AD-A280 993

NAVAL POSTGRADUATE SCHOOL
MONTEREY, CALIFORNIA

THESIS

ANALYSIS OF THE FIRST SUCCESSFUL FLIGHT OF
GPS ABOARD THE SPACE SHUTTLE

by

Stephen Paul Rehwald, Jr.
and
Carolyn Louise Tyler

March 1994

Thesis Advisor: Randy L. Wight

DTIC QUALITY INSPECTED 3

94-20401

94 7 5 112

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS

2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	Approved for public release; distribution is unlimited

4. PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S)

6a. NAME OF PERFORMING ORGANIZATION Naval Postgraduate School	6b. OFFICE SYMBOL (if applicable) Code 31	7a. NAME OF MONITORING ORGANIZATION Naval Postgraduate School

6c. ADDRESS (City, State, and ZIP Code) Monterey, CA 93943-5000	7b. ADDRESS (City, State, and ZIP Code) Monterey, CA 93943-5000

8a. NAME OF FUNDING/SPONSORING ORGANIZATION	8b. OFFICE SYMBOL (if applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER

8c. ADDRESS (City, State, and ZIP Code)	10. SOURCE OF FUNDING NUMBERS			
	PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)
ANALYSIS OF THE FIRST SUCCESSFUL FLIGHT OF GPS ABOARD THE SPACE SHUTTLE

12. PERSONAL AUTHOR(S)
Rehwald, Stephen P., and Tyler, Carolyn L.

13a. TYPE OF REPORT Master's Thesis	13b. TIME COVERED FROM 02/93 To 03/94	14. DATE OF REPORT (Year, Month, Day) March 1994	15. PAGE COUNT 111

16. SUPPLEMENTARY NOTATION   The views expressed in this thesis are those of the authors and do not reflect the official policy or position of the Department of Defense or the United States Government.

17.	COSATI CODES		18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Global Positioning System, GPS, Space Shuttle, Kalman Filtering, Coordinte Transformation, State Vector.

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

A Trimble Advanced Navigation Sensor (TANS) Quadrex Global Positioning System (GPS) receiver processing unit and three antenna/preamplifier assemblies were flown aboard Space Shuttle *Discovery*, STS-51, as part of DTO 700-6, GPS On-orbit Demonstration (GOOD). The experiment was designed to quantify advantages and identify potential problem areas for Space Shuttle GPS operations using a low cost, commercial, space configured, GPS receiver. GPS data, including position, velocity, time, health, and status information were recorded during the mission. Following the mission, a reference trajectory was generated by NASA Johnson Space Center through post-processing of the Orbiter's on board navigation state. The recorded GPS data has been analyzed and compared to the reference trajectory to evaluate the navigational performance of the receiver. Additionally, post-flight filtering of the GPS data has been performed in order to determine whether a significant increase in performance may be obtained through filtering.

DTIC QUALITY INSPECTED 3

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT [X] UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED	
22a. NAME OF RESPONSIBLE INDIVIDUAL Randy L. Wight	22b. TELEPHONE (Include Area Code) (408) 646-2491	22c. OFFICE SYMBOL SP/Wt

ANALYSIS OF THE FIRST SUCCESSFUL FLIGHT OF GPS
ABOARD THE SPACE SHUTTLE

by

Stephen Paul Rehwald, Jr.
Lieutenant, United States Navy
B.S., United States Naval Academy, 1986

and

Carolyn Louise Tyler
Lieutenant, United States Navy
B.S., Mary Washington College, 1986

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ASTRONAUTICAL ENGINEERING
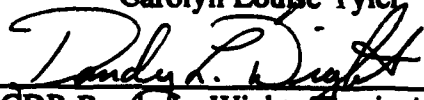
from the

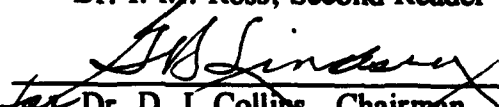NAVAL POSTGRADUATE SCHOOL
March 1994

Authors: _____
Stephen Paul Rehwald, Jr.

_____
Carolyn Louise Tyler

Approved By: _____
CDR Randy L. Wight, Thesis Advisor

_____
Dr. I. M. Ross, Second Reader

_____
Dr. D. J. Collins, Chairman,
Department of Aeronautical and Astronautical Engineering

ii

ABSTRACT

A Trimble Advanced Navigation Sensor (TANS) Quadrex Global Positioning System (GPS) receiver processing unit and three antenna/preamplifier assemblies were flown aboard Space Shuttle *Discovery*, STS-51, as part of DTO 700-6, GPS On-orbit Demonstration (GOOD). The experiment was designed to quantify advantages and identify potential problem areas for Space Shuttle GPS operations using a low cost, commercial, space configured, GPS receiver. GPS data, including position, velocity, time, health, and status information were recorded during the mission. Following the mission, a reference trajectory was generated by NASA Johnson Space Center through post-processing of the Orbiter's on board navigation state. The recorded GPS data has been analyzed and compared to the reference trajectory to evaluate the navigational performance of the receiver. Additionally, post-flight filtering of the GPS data has been performed in order to determine whether a significant increase in performance may be obtained through filtering.

Accession For	
NTIS GRA&I	☑
DTIC TAB	☐
Unannounced	☐
Justification	

By	
Distribution/	
Availability Codes	

Dist	Avail and/or Special
A-1	

iii

TABLE OF CONTENTS

TABLE OF ABBREVIATIONS

BET -- Best Estimate of Trajectory

C/A-Code -- Clear Acquisition Code

CTS -- Conventional Terrestrial System

DOD -- Department of Defense

DOP -- Dilution of Precision

DTO -- Detailed Test Objective

GDOP -- Geometric Dilution of Precision

GOOD -- GPS On Orbit Demonstration

GMST -- Greenwich Mean Sidereal Time

GMT -- Greenwich Mean Time

GPS -- Global Positioning System

IAU -- International Astronomical Union

IMU -- Inertial Measurement Unit

JAM -- Junction Adapter Module

MSBLS -- Microwave Scan Beam Landing System

M50 -- Aries-mean-of-1950

NASA -- National Aeronautics and Space Administration

OMS -- Orbital Maneuvering System

ORFEUS -- Orbiting Retrievable Far and Extreme Ultraviolet Spectrometer

PCMMU -- Pulse Code Modulation Master Unit

P-Code -- Precision Code

PDOP -- Position Dilution of Precision

PGSC -- Payload and General Support Computer

PPS -- Precise Positioning Service

RCS -- Reaction Control System

RF -- Radio Frequency

SA -- Selective Availability

SPAS -- Shuttle Palette Satellite

SPS -- Standard Positioning Service

STS -- Space Transportation System

TACAN -- Tactical Air Navigation

TANS -- Trimble Advanced Navigation Sensor

TDOP -- Time Dilution of Precision

TDRS -- Tracking and Data Relay Satellite

UTC -- Coordinated Universal Time

UT1 -- Universal Time 1

WGS84 -- World Geodetic System 1984

ACKNOWLEDGEMENT

I. INTRODUCTION

This thesis investigated the performance of a low cost, commercial, space configured, Global Positioning System (GPS) receiver[1] flown aboard Space Shuttle *Discovery*, STS-51, as part of DTO 0700-6,[2] GPS On-orbit Demonstration (GOOD). The DTO was sponsored and funded by NASA Johnson Space Center, and developed with the support of NASA contractors, and students, including the authors, from the Naval Postgraduate School. Data recorded during the mission was analyzed to evaluate navigational performance of the receiver. Additionally, post-flight filtering of this data was performed in order to determine whether a significant increase in performance could be obtained through filtering.

A. BACKGROUND

1. Space Shuttle Orbiter Baseline Navigation System

A wide variety of equipment is employed in the Orbiter's baseline navigation system. All navigation sensor information is supplied to a six-state suboptimal Kalman filter, which provides the navigation functions with three position and three velocity states. The three position states are the coordinates specifying the Orbiter's position vector in the Aries-mean-of-1950 (M50) Cartesian coordinate system.[3] Likewise, the three velocity states define the Orbiter's velocity vector in the M50 system.

---

[1] Trimble Advanced Navigation Sensor (TANS) Quadrex GPS Receiver Processor Unit.

[2] Detailed Test Objective Number 0700-6.

[3] The M50 system is defined in NASA Technical Memorandum X-58153, October 1974.

During the ascent phase of a mission, the Inertial Measurement Unit (IMU) is the primary sensor, providing attitude and acceleration data to the Kalman filter. This data is augmented by ground based C-band radar-tracking information uplinked over an S-band communication link. During the on-orbit coasting phase of a mission, the IMU provides attitude data, and acceleration data from Orbital Maneuvering System (OMS) burns. Accelerations falling below the IMU threshold arise from opposing Reaction Control System (RCS) thrusters that do not form a perfect couple (vernier effect), and from external venting of gasses and waste products. These unaccounted for accelerations result in steadily increasing navigational error.

While on-orbit, the ground continues to track the Orbiter using ground-based C-band radar. Two-way Doppler tone ranging over S-band and Ku-band communication links, either direct or via the Tracking and Data Relay Satellite (TDRS) system, provides additional tracking capability. When the Orbiter's navigational state is observed to deviate from the ground based tracking trajectory by a pre-defined, mission-dependent amount, a new state vector consisting of three position states, three velocity states, and a time tag is uplinked by Mission Control. At various times during a mission (prior to rendezvous and deorbit burn), an IMU alignment may be performed using an on board star-tracker to correct attitude error caused by gyro drift.

During the re-entry phase of a mission, IMU data is augmented with drag modeling data from 250,000 feet down. As the Orbiter passes through the ionosphere, all radio-navigation and communication signals are blacked out for a period of time. Upon exit from blackout, contact is first established by C-band tracking radar. A state vector can be uplinked as soon as S-band communication is regained. Subsequently, the Orbiter can receive L-band Tactical Air Navigation (TACAN) station signals and begin area navigation, combining TACAN range and bearing data with barometric (30,000-2,500 feet) and radar (2,500 feet down) altimeter measurements. Final approach and

2

landing are accomplished with a microwave scan beam landing system (MSBLS), beginning normally at 10,000 feet, 10 nautical miles downrange from touchdown.

2.   Space Shuttle Orbiter GPS Navigation System

Pursuant to a study contract commissioned by NASA, Rockwell International's Space Systems group conducted a design and integration study of a GPS-based primary navigation system for the Orbiter in the late 1970's. (Van Leeuwen et al, 1979, pp. 118-135) The study demonstrated that the use of on board satellite GPS receivers for precise orbit determination was clearly feasible, and expected the improved navigation capabilities to yield significant operational benefits. The study concluded the GPS-based system to be a technically sound and cost-effective proposition. Based on the study, plans were laid to install a GPS navigation system in all Shuttle orbiters beginning in the early 1980's, with follow-on goals of deleting certain equipment from the baseline navigation system. Within about two years, however, the decision to install GPS was reversed in favor of continuing with the baseline navigation configuration. Certain GPS provisions, notably antennas, cabling, and bulkhead feedthroughs, were nevertheless retained, and currently exist on all Orbiter vehicles. (Saunders, 1994, pp. 1-13)

The issue of GPS installation in the Orbiter fleet surfaced again in the early 1990's. Renewed interest was motivated by the planned phase out of TACAN stations. Since TACAN was used as the Orbiter's primary navigation aid following exit from blackout, through MSBLS acquisition, NASA considered suitable alternatives. Looking at the direction the Department of Defense (DOD) and the Federal Aviation Administration were heading in, GPS was chosen as the replacement for TACAN. A developmental test for the Orbiter GPS navigation system flew aboard STS-61 in December 1993, and the system is presently expected to be operational in 1996. (Kachmar et. al., 1993, pp. 313-326)

3. GPS On-orbit Demonstration

In mid-1992, with the foundation for installing an Orbiter GPS navigation system laid, the crew of STS-51 conceived the GOOD DTO as a low cost pathfinder project, to look at GPS in orbit, to quantify advantages, and identify potential problem areas for Space Shuttle operations. Data from the DTO could then be used to complement the more highly integrated GPS Development Flight Test. Since STS-51 would carry another payload with its own GPS receiver, the Orbiting Retrievable Far and Extreme Ultraviolet Spectrometer-Shuttle Palette Satellite (ORFEUS-SPAS), the DTO would also permit the evaluation of relative GPS[1]. Successful utilization of GPS on the Orbiter could show benefits for use on other programs, such as Space Station, or for use as a utility with other primary and secondary payloads, which require precise location and timing information. Initially, goals of this experiment were as follows:

- Evaluate receiver performance in orbit by comparing its state vector to that determined by ground tracking and Orbiter IMU's.
- Evaluate the number and location of GPS antennas required to provide best navigation solutions for flight deck experiment applications.
- Determine the quality of GPS data received during on-orbit operations by collecting GPS health data.
- Evaluate the accuracy of relative GPS, using GPS receivers both in the crew cabin and on ORFEUS-SPAS, with Orbiter radar and laser rangefinders as a reference.
- Evaluate postflight the accuracy of relative GPS using data from Orbiter and SPAS GPS receivers.

One of the computer displays developed for this DTO showed the magnitude of the position difference between the Orbiter GPS and Orbiter IMU based state vectors versus time.

---

[1] Aspects of relative GPS are covered in the thesis "Theoretical Basis for State Vector Comparison, Relative Position Display, and Relative Position/Rendezvous Prediction" by LT Lester Makepeace, and the thesis "NPS State Vector Analysis and Relative Motion Plotting Software for STS-51" by LT Lee Barker.

Since error in the IMU based state vector increased with time, the root sum square (RSS) difference (delta) between the GPS state vector and IMU state vector was expected to increase with time. This difference was expected to collapse to zero when Mission Control uplinked a new state vector based on ground tracking. This behavior was first observed in orbit on flight day three, when a 20,000 ft. delta collapsed to about 300 feet following an update. An illustration of a similar event on the computer display is shown in Figure 1 (the x-axis represents time, the y-axis represents RSS difference). The close correlation between expected behavior and actual behavior indicated the GPS position solution to be near truth.
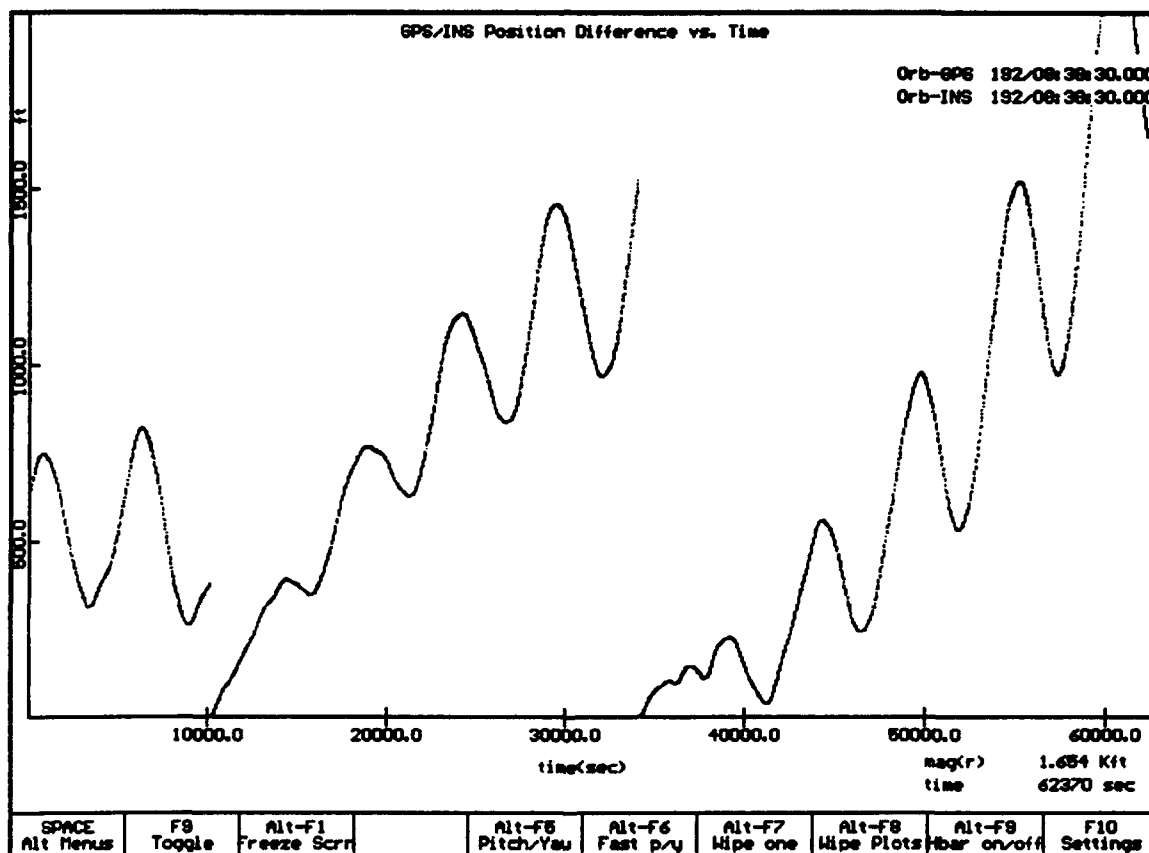
Figure 1: State Vector Differences vs. Time

B. GPS OVERVIEW

The GPS constellation consists of 21 operational satellites, and three active spares, distributed in six orbital planes with three or four operational satellites in each plane. The ascending nodes of each plane are separated by 60° intervals, and each plane has an inclination relative to the equator of 55°. The satellites orbit at an altitude of 20,200 km, with a corresponding period of 12 hours. In comparison, the Space Shuttle orbits at an altitude of approximately 300 km, with a corresponding period of 1½ hours. The satellites are positioned so that a minimum of five will normally be observable to a user located anywhere on earth.

The satellites transmit on two frequencies: L1 = 1575.42 MHz and L2 = 1227.6 MHz. The satellites transmit their signals using spread spectru a techniques employing two different spreading functions: a 1.023 MHz coarse/acquisition (C/A) code on L1 only and a 10.23 MHz precision (P) code on both L1 and L2. Both P-code and C/A-code enable a receiver to determine the range between the satellite and the user. Superimposed on both the P-code and the C/A-code is the NAVIGATION message (NAV-msg), containing satellite ephemeris data, atmospheric propagation correction data, and satellite clock-bias information. The TANS Quadrex GPS receiver flown on STS-51 utilizes only the C/A-code on the L1 frequency carrier.

Two levels of navigation are provided by the GPS; these are Precise Positioning Service (PPS) and Standard Positioning Service (SPS). The PPS is a highly accurate positioning, velocity, and timing service which is made available only to authorized users through cryptographic keys. The SPS is a less accurate positioning and timing service which is available to all GPS users. The TANS Quadrex GPS receiver flown on STS-51 is an SPS receiver. In the future, receivers to be installed as part of the Orbiter GPS navigation system will be ᴾPS units. (Kachmar, et.al., 1993, pp. 313-326)

The SPS is specified to provide a 100 meter (95% confidence) horizontal accuracy to any GPS user during peacetime. This is approximately equal to 156 meters three-dimensional (3-D) (95% confidence) accuracy. SPS receivers can achieve approximately 337 nanosecond (95% confidence) Coordinated Universal Time (UTC) time transfer accuracy. The SPS is primarily intended for civilian purposes, although it has many peacetime military uses as well. The SPS horizontal accuracy specification include peacetime degradation of Selective Availability (SA) which is the dominant SPS source.[1] The SA position error distribution resembles a Gaussian distribution with a long-term mean of zero. The SPS peacetime velocity degradation due to SA is classified.

The ranging codes broadcast by the satellites enable a GPS receiver to measure the transit time of the signals and thereby determine the range between a satellite and the user. The NAV-msg enables a receiver to calculate the position of each satellite at the time of transmission of the signal. Four satellites are normally required to be simultaneously "in view" of the receiver for 3-D positioning purposes. This allows the user 3-D position coordinates and the user clock offset to be calculated from the satellite range and position data. Treating the user clock offset as an unknown eliminates the requirement for users to be equipped with precision clocks. Less than four satellites can be used if the user altitude or system time is precisely known.

When the receiver has acquired the satellite signals from four GPS satellites, achieved carrier and code tracking, and has read the NAV-msg, the GPS receiver is ready to start navigating. The GPS receiver normally updates its pseudoranges and relative velocities once every second. The measurements are termed pseudorange because

---

[1]SPS did not routinely meet accuracy specs while the GPS system was undergoing test and verification by the DOD, prior to December, 1993. SA effects were often varied to further degrade accuracy.

the clock offset of a GPS receiver introduces a bias to the true range of the satellite. The GPS receiver must know the GPS system time very accurately, because the satellite signals contain the time of transmission from the satellite in GPS time. The difference in time between the signal leaving the satellite and arriving at the GPS receiver antenna is directly proportional to the range between the satellite and the GPS receiver, so it is of the utmost importance that the same time reference is used by both the GPS satellites and the GPS receiver.

The GPS satellites carry two rubidium, and two cesium atomic frequency standards. However, the GPS receiver is not required to have a high accuracy clock such as an atomic time standard. Instead, a crystal oscillator is used and the GPS receiver corrects its offset from GPS system time by making four pseudorange measurements. The GPS receiver can use the four pseudoranges to solve four simultaneous equations with four unknowns. The position equations are shown in Figure 2. When the four equations are solved, the GPS receiver has estimates of its position and GPS system time. The GPS receiver velocity is calculated using the same types of equations, using relative velocities instead of pseudoranges. GPS receivers perform most calculations using an earth-centered earth-fixed coordinate system. They then convert to an earth model defined by the World Geodetic System 1984 (WGS 84). WGS 84 is a very precise model that provides a common grid system for transformations into other coordinate systems or map datums.

Satellite coverage, as measured at the user antenna, can be affected by physical obstructions, vehicle maneuvering or aspect, and basic receiver design. It therefore cannot be categorized as a GPS system requirement or specification. Coverage is defined by the orbits of the active satellites. The orbits determine the geometric relationships between the satellites and the user, which the user measures as Position Dilution of Precision (PDOP). Since the geometric relationships continuously change as the satellites

8

$$R_1 = c\Delta t_1 \qquad\qquad (x_1 - U_x)^2 + (y_1 - U_y)^2 + (z_1 - U_z)^2 = (R_1 - C_b)^2$$

$$R_2 = c\Delta t_2 \qquad\qquad (x_2 - U_x)^2 + (y_2 - U_y)^2 + (z_2 - U_z)^2 = (R_2 - C_b)^2$$

$$R_3 = c\Delta t_3 \qquad\qquad (x_3 - U_x)^2 + (y_3 - U_y)^2 + (z_3 - U_z)^2 = (R_3 - C_b)^2$$

$$R_4 = c\Delta t_4 \qquad\qquad (x_4 - U_x)^2 + (y_4 - U_y)^2 + (z_4 - U_z)^2 = (R_4 - C_b)^2$$

R_i = pseudorange

c = speed of light

Δt_i = time difference between signal leaving the satellite and arriving at the receiver

x_i, y_i, z_i = satellite position

U_x, U_y, U_z = receiver antenna position

C_b = receiver clock bias

Figure 2: GPS Position Equations

move round their orbits, so does the user's value of PDOP. PDOP is defined as the square root of the variances of the position errors (PDOP = $(\sigma_x^2 + \sigma_y^2 + \sigma_z^2)^{1/2}$), and its effects are illustrated in Figure 3.

C. SUMMARY

The GOOD DTO was designed to display real time GPS position and velocity data to the astronaut crew while on-orbit, and to record this data on hard disk for post-flight comparison with a ground generated Best Estimate of Trajectory (BET). It was also designed to display real time state vector differences between the following:

- TANS GPS and Orbiter IMU
- TANS GPS and SPAS GPS
- Orbiter IMU and SPAS GPS

The state vector differences between SPAS GPS and either the Orbiter IMU or TANS GPS could then be compared against on-board radar and laser rangefinder results.

9
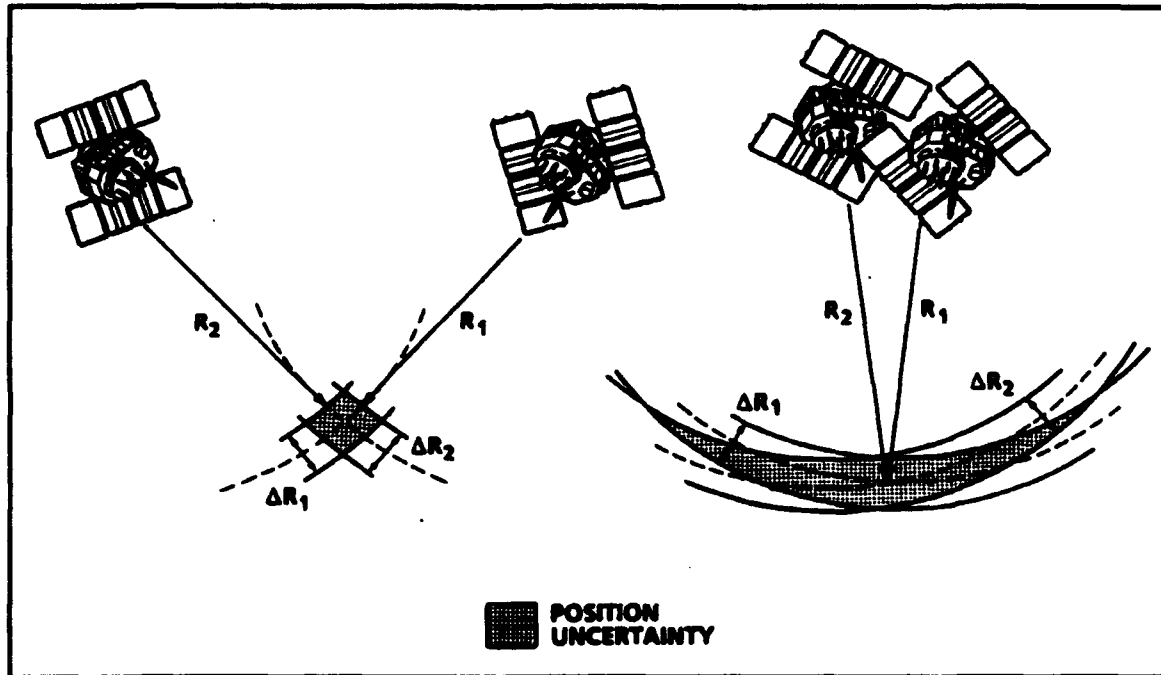
Figure 3: Effects of PDOP

1. State Vector Differencing

Three prerequisites had to be satisfied before two state vectors could be compared or differenced. These were as follows:

- Position and velocity elements were in the same frame of reference.
- Time elements were in the same time scale.
- Time elements were exactly matched.

Since the Shuttle navigation system used the M50 Earth-centered Inertial frame of reference, and the GPS receivers utilized the WGS 84 Earth-centered Earth-fixed frame of reference, GPS state vectors were rotated from the WGS 84 reference frame to the M50 reference frame prior to comparison with the Orbiter IMU state vector. This coordinate transformation will be addressed at length in a later chapter.

Since the Shuttle navigation system used Greenwich Mean Time (GMT), and the GPS system used GPS time, GPS state vector times were adjusted to GMT prior to

comparison with Orbiter IMU state vectors. The Orbiter's clock was set according to the National Bureau of Standards UTC standard, making the GMT time scale equivalent to the UTC time scale in this application. The difference between UTC and GPS time was transmitted in the NAV-msg by GPS satellites, and this value was used for the adjustment.

Since the state vectors being compared were produced by independent systems, in general, the time elements did not match. The GOOD DTO utilized Cowell's method to "propagate" the earlier state vector forward in time until its time element matched the time element of the state vector it was being compared to. Although the difference between time elements was typically less than a second, it was significant at orbital velocities of several kilometers per second. Quick and accurate propagation of states is an active area of research, but will not be addressed further in these pages.[1]

2. State Vector Filtering

Both the Orbiter IMU, and the SPAS GPS state vector outputs were filtered in order to smooth the outputs over time, and to improve accuracy. The TANS GPS statevector outputs, however, did not undergo filtering. Though use of a Kalman filter to smooth the output would likely improve the TANS accuracy, a filter was not implemented for the GOOD DTO due to time constraints. A Kalman filter has since been designed for use with the TANS and will be addressed in a later chapter.

---

[1] See the master's thesis of LT Lester Makepeace for a discussion of propagation, and an alternative to Cowell's method for this application.

11

II. EXPERIMENT DESCRIPTION

A. INTRODUCTION

The test objectives for this experiment were to demonstrate GPS on-orbit perfor-
mance at a relatively low cost. To meet this objective, NASA needed to:

- minimize interfaces to the Orbiter;
- use "off-the-shelf" GPS technology;
- designate this flight test as a non-critical (Detailed Test Objective) DTO; and,
- limit hardware/software certification and qualification to ensuring crew safety.

Other objectives for this flight included collecting on-orbit GPS data to be pro-
cessed post flight, demonstrating the GPS performance to STS-51 crew real time, and
evaluating potential future use for this hardware and software.

The GOOD (GPS On-Orbit Demonstration) software was developed for use on a
GRID 386 laptop computer operating at 10 MHz. The desire was to provide the crew
with the capability to command and control the GPS receiver, and to display and record
GPS data for real time and postflight analysis. In February 1993, when our Naval Post-
graduate School team (LT Lee Barker, LT Les Makepeace, LT Steve Rehwald, and LT
Carolyn Tyler) arrived at NASA, Johnson Space Center, the flight hardware had already
been selected. The software used to interface with the TANS GPS receiver was being
fine-tuned to NASA's needs. Software used to provide real time analysis of the GPS data
had not been completed.

B. LEVEL I: HARDWARE

The hardware selected by the NASA team consisted of the TANS receiver and its
associated preamplifier, cabling, and three antennas. Foam spacers were used to keep

the antennas away from damaging the sensitive ultraviolet radiation protective coating on the Orbiter's windows. Radio Frequency (RF) absorbers were placed behind the antennas to minimize the RF disturbances in the crew cabin. Finally, the entire assembly was attached to the window using velcro straps. As a safety requirement, a JAM (Junction Adapter Module) was designed and built. The JAM was the only electrical interface to the Orbiter and was required to protect the Orbiter from any adverse electrical behavior generated by the GPS experiment. The hardware set up used aboard Discovery is shown in Figure 4.
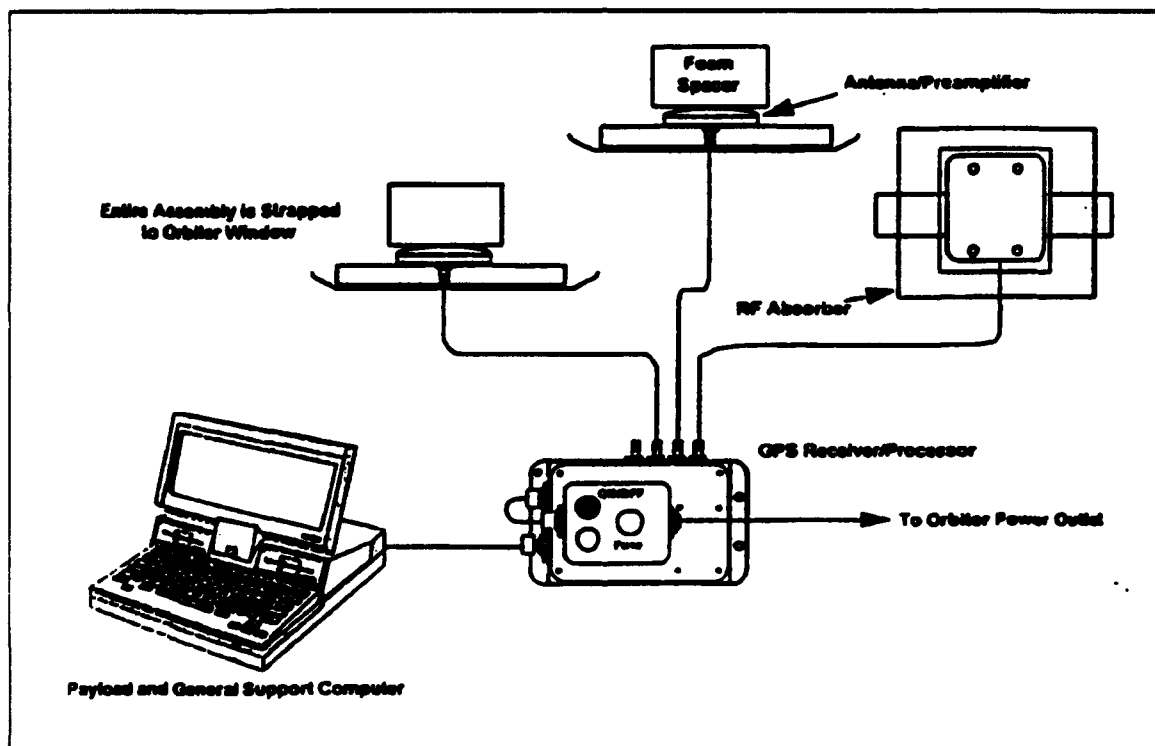
Figure 4: Hardware Setup

A more specific description of the TANS receiver used in this experiment is listed in Table 1.

Table 1: TANS RECEIVER DESCRIPTION

TANS Description	Specification
Code / Carrier Tracked	C/A code, L1
Channels	6
Antenna input signals	up to 4
Position Accuracy	25 meters (SEP) without SA
	100 meters (2DRMS) with SA
Velocity Accuracy	0.2 m/s without SA
	classified with SA
Time Accuracy	1 microsecond of UTC
Dimensions:	
- Receiver / Antenna	5"x9.5"x2.2" / 3.75"x4"x0.54"
Prime Power	3.5 Watts @ 28 VDC
Weight: Receiver / Antenna	3.5 lbs / 0.4 lbs
Dynamic Capability:	
- Velocity	8000 m/s
- Acceleration	4 g's
- Jerk	2 g's/sec
Data Interface	RS 422 dual channel
Temperature:	
- Operating / Non-Operating	-40 to 70 / -55 to 85 degrees C
Altitude	1100 nautical miles
Vibration	0.04 g /Hz, 100 to 1100 Hz
Shock	40 g / 11 ms, 75 g / 6 ms
Humidity	100% condensing

The TANS receiver was chosen because it was space-configured and it met the main objective of being available at a relatively low cost. A disadvantage to using this receiver was that it had Standard Positioning Service (SPS) capability and not the preferred Precise Positioning Service (PPS). The latter capability would have increased cost, slowed progress and put special restrictions on this experiment which would have prevented it from making STS-51's scheduled flight deadline. In the future, NASA intends to use the PPS capability which improves accuracy by 100 to 156 meters as compared to the SPS receiver. Another disadvantage to using this receiver was its use of a deterministic point solution design instead of a filter. If the receiver had incorporated a Kalman Filter as a part of its design, the processor unit would perform calculations based on a filtering design and not based on user-selected specifications. In this flight test, as an example, NASA chose a 3-D Manual selection, an option the TANS provides the user. If the 3-D Manual is on, a three dimensional solution will not be calculated unless four satellites are in view and meet certain requirements.

There were several settings made to the TANS which kept this receiver from being tested in its best configuration. The optimum TANS receiver/antenna performance was found to occur when antennas were placed on an unobstructed flat surface, looking straight up into the GPS constellation. Unfortunately, the Orbiter does not fly in an attitude or provide a window set up to facilitate such an optimum antenna placement, but rather the Shuttle flies in a left, right, or both wings down attitude (payload bay down towards the earth) with small restrictive windows. As one might guess, the worst position for the receiver was when the Orbiter was in the latter position, with all three antennas facing mostly away from the GPS satellites. Due to this major drawback, NASA made special exceptions to important settings like Position Dilution of Precision (PDOP), Signal Level Mask and Elevation Angle.

The PDOP mask, a user-specified setting, should normally be set to a maximum of six. This setting ensures that four satellites are in a satisfactory geometrical position. NASA set the PDOP at 20 because there was concern that a setting of six might be too restrictive, due to limited viewing angles and Orbiter attitudes, which would cause the receiver to make very few solutions.

NASA set the Signal Level Mask specification at five AMU's (a linearized measurement of noise, usually expressed in decibels) which is a very low setting. Nominal levels are in the 15 to 25 range. If this setting had been lower, more cockpit noise would have interfered. As it was, the PGSC's (the other laptops in use) were picked up by the antennas and tracked by the receiver, even at levels below five AMU's. Trimble, since this experiment, has modified the TANS software package to correct for this problem. The need for this low setting was apparent after looking at the way a satellite was tracked. In order to track the satellite as long as possible, the signal cut off had to be low. Also, the elevation angle was required to be low to enhance tracking, and in this case, NASA set the angle to zero. This encouraged error due to multipath effects and the tendency of the antenna to pick up competing signals (i.e. the PGSC) when tracking the satellites through low angles. In Chapter V, the above error inducing settings will be covered in greater detail.

C. LEVEL I AND II: SOFTWARE OVERVIEW

Software was classified as either Level I or Level II. These categories reflected not only their level of importance, but the minimum requirements needed for the experiment to fly aboard the shuttle.

At the important but basic first level, this experiment required the ability to operate the TANS receiver and display certain data to a PGSC (a GRID 386/10) screen. It had to store state vectors and engineering data to files, with the ability to downlink this

data. The downlink capability was important should a situation arise when the astronaut crew required troubleshooting assistance from ground experts during the flight. .

At the more advanced level, Level II, NASA desired some additional capability, such as the ability to input an Orbiter state vector either manually or automatically to compare the GPS state vector measurement to the Orbiter's. NASA also wanted to compare the TANS GPS measurements with another GPS receiver used on the STS-51 payload, ORFEUS-SPAS. One of the primary missions for STS-51 was to carry this German-made satellite into space, release it to operate independently for several days, and then rendezvous to recover it prior to returning home. The NASA engineers saw this as an opportunity to study the relative GPS technique.

At a minimum, NASA wanted to collect the GPS data via the TANS receiver for postflight studies. Ultimately, even after both Levels I and II were fully developed, this GOOD test was only an experiment (or DTO). It was to be operated by the STS-51 crew on a not-to-interfere basis, only. An example of interference occurred during the rendezvous with the ORFEUS-SPAS, which was one of the specified phases of flight to record information for postflight study. The TANS experiment could not be run because the antennas, strapped in the windows, adversely blocked the crew's view for rendezvous and as a safety of flight concern interfered with the crew and their duties.

D. LEVEL I: SPECIFICS

Available in the Level I software were four interface displays. One display showed the user a current TANS configuration set-up. A second display was used to send commands and requests to the TANS receiver. A third showed the Orbiter's location on a world map, and the forth display showed data for crew monitoring. This data consisted of six rows of information for six channels and their related channel ID, satellite ID, acquisition flag, ephemeris flag, azimuth, elevation, and doppler.

Also provided to the user was position and velocity in two different coordinate systems. The TANS receiver made measurement calculations using WGS-84 . Its output for position and velocity in this frame was either in cartesian coordinates or latitude, longitude, and altitude. A non-trivial coordinate transformation was performed on the WGS-84 position and velocity to translate them into one of the key reference frames used by the Orbiter, M50.

E.   LEVEL II: SPECIFICS

The Level II software provided real-time graphical displays of the TANS or ORFEUS-SPAS GPS state vector measurements and GPS/Orbiter state vector comparisons. Figure 5 demonstrates the entire GPS configuration in a block diagram.
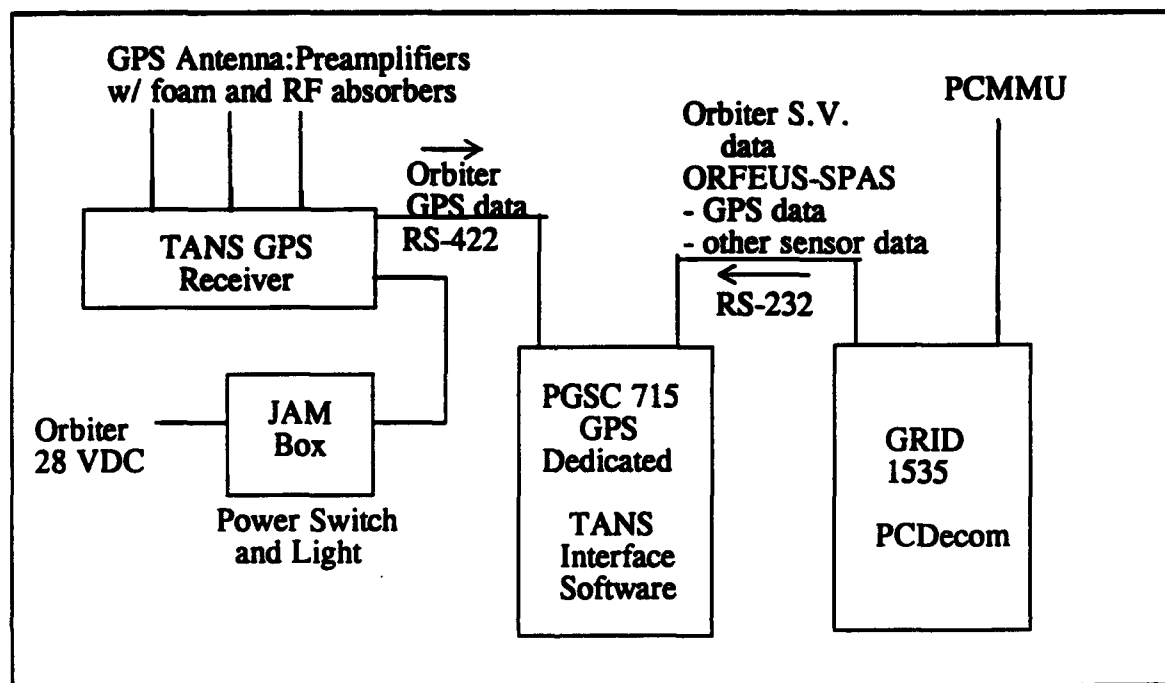
Figure 5: Experiment Block Diagram

In words, the TANS GPS information traveled from the receiver via the RS-422 connection to the PGSC 715 (the laptop dedicated to GPS). The Orbiter state vector and

18

ORFEUS-SPAS GPS data traveled from the PCMMU via another GRID 1535 laptop running the PCDecom program through the RS-232 connection to the PGSC 715 to be manipulated in the Level II code. Part of the manipulation designs were to use the Orbiter's GPS (from TANS) and ORFEUS-SPAS GPS state vectors in a rendezvous program.

Much of the Level II code was written by the NPS team, with major guidance and assistance from a computer programming wizard, and author of the PCDecom program, Mr. Tom Silva. Greater detail about the flight code and mathematical derivations is supplied in the theses written by LT Lee Barker and LT Les Makepeace.

III. STATE VECTOR DIFFERENCING

A. DATA ANALYSIS

1. Reference Trajectory

Following the STS-51 mission, trajectory data for the GOOD DTO was compiled by the Shuttle Navigation group at NASA Johnson Space Center. In lieu of a true Best Estimate of Trajectory, a reference trajectory, generated by propagating between real-time navigation solutions as computed during the mission, was created. The resulting trajectory had an accuracy of 225 meters and 0.15 meters per second in total position and velocity. This accuracy was deemed sufficient given the less than optimal test conditions of the DTO.

Orbiter state vectors, whose time elements exactly matched those of GPS state vectors[1] were computed by interpolating within the reference trajectory using a cubic spline technique. This method introduced an estimated five meters of additional position error into the reference data.

2. GPS Data

Most of the data output by the TANS GPS receiver during the GOOD DTO was recorded for postflight analysis. A summary of recorded data is shown in Table 2. Due to the sheer volume of data (over 15MB) recorded, a representative sample was analyzed. Of approximately 2500 recorded GPS fixes, 369 (roughly 15%) were chosen from three different time periods for analysis. Selected double precision XYZ, and Velocity XYZ ECEF binary formatted data packets were reformatted into ASCII text GPS state vectors using the TANSPOST program supplied by NASA.

---

[1]TANS GPS position and velocity solution times-of-fix were always matched.

Table 2: GPS DATA OUTPUT

Packet Description	Packet Size (bytes)	Rate
Double Precision XYZ	40	0.33 Hz
Double Precision LLA	40	0.33 Hz
Velocity, XYZ ECEF	24	0.33 Hz
Raw measurement output	30-36	0.33 Hz
Satellite Tracking Status	28	0.33 Hz
Satellite Selection	25	0.033 Hz
Report Operating Parameters	21	once+any change
Report Control Options	8	once+any change
Health of TANS	6	0.033 Hz
Almanac Information	70	On request
Almanac Health Page, Time & Week	41	On request
Ionospheric Parameters	44	On request
UTC Parameters	43	On request
Oscillator Offset	8	On request
Satellite Ephemeris Status	171	On request

3. Data Reduction

Before a comparison could be made between the Orbiter reference state vectors and the TANS GPS state vectors, a coordinate rotation was required to transform the GPS state vectors from the WGS 84 Earth-centered Earth-fixed frame of reference to the M50 mean inertial frame of reference. Since the standard epoch of B1950.0, upon which M50 is based, was superseded in 1976 by the standard epoch of J2000.0, very little information appears in the current literature for working with the M50 system. Instead, direction is given for making transformations to and from the J2000.0 mean

21

inertial coordinate system. While NASA continues to utilize older transformation theory to convert Earth-centered Earth-fixed coordinates directly to M50, the authors chose to make the transformation first into the J2000.0 mean inertial system, and then into the M50 mean inertial system using a constant transformation matrix published by NASA Jet Propulsion Laboratory (Standish, 1982, pp. 297-302) to relate the M50 and J2000.0 systems. The transformation into J2000.0 allowed the use of the newer 1980 IAU theory of nutation for improved accuracy.

4. Coordinate Transformations

The WGS 84 system is more accurately referred to as a geopotential model that has adopted the Conventional Terrestrial System (CTS) (1984.0), defined by the Bureau International de l'Heure, as its reference frame. Earth's center of mass is the origin of the CTS, as well as the M50 and J2000.0 systems. The Z-axis of the CTS is known as the Conventional Terrestrial Pole (CTP). The X-axis of the CTS is the Zero meridian, and is used to derive Universal Time, specifically UT1, in the same way the Greenwich meridian is used to derive GMT. The Y-axis completes a right-handed coordinate system. (DMA-TR-8350.2, 1987, p. 2-1)

Transformation of WGS 84 coordinates into M50 coordinates requires two 3x3 rotation matrices, each computed for the time of the state vector being transformed. Transformation of position vectors actually requires only a single rotation matrix. Subsequent discussion will refer to this matrix as "matrix 1." Transformation of velocity vectors, however, requires a second matrix to account for the rate of change within matrix 1. This matrix will be referred to as "matrix 2."

Matrix 1 is actually the product of five separate 3x3 matrices, premultiplied to form a single matrix. The letters A, B, C, P, and M will be used to denote these five matrices. A applies two rotations for polar motion. B applies a single rotation for sidereal time (Earth rotation). C applies three rotations for astronomic nutation. P applies

22

three rotations for general precession, and M applies three rotations for standard epoch conversion. All matrices with the exception of M are time varying. The M50 position vector is the product of the transpose of matrix 1, and the WGS 84 position vector.

Matrix 2 is also the product of five separate 3x3 matrices. Four of the matrices used to compute matrix 1, A, C, P, and M, are also used for matrix 2. Unlike the other time varying matrices, the rate of change of the B matrix is significant; so another matrix, denoted by $\dot{B}$, replaces the B matrix in matrix 2. $\dot{B}$ is the rate of change of the B matrix. The M50 velocity vector is computed in three steps. In step one, the product of the transpose of matrix 2 and the WGS 84 position vector is computed. In step two, the product of the transpose of matrix 1 and the WGS 84 velocity vector is computed. In step three, the two resulting vectors are added vectorially to form the M50 velocity vector. The transformation methodology for position coordinates is shown in Equation 1 and the transformation methodology for velocity coordinates is shown in Equation 2.

$$\left[xyz\right]_{M50}^{T} = \left[ABCPM\right]^{T}\left[xyz\right]_{WGS-84}^{T} \qquad \text{(Eq: 1)}$$

$$\left[\dot{x}\dot{y}\dot{z}\right]_{M50}^{T} = \left[A\dot{B}CPM\right]^{T}\left[xyz\right]_{WGS-84}^{T} + \left[ABCPM\right]^{T}\left[\dot{x}\dot{y}\dot{z}\right]_{WGS-84}^{T} \qquad \text{(Eq: 2)}$$

Methods for computing the A, B, $\dot{B}$, C, and P matrices were taken from The Astronomical Almanac, The Explanatory Supplement to the Astronomical Almanac, and Defense Mapping Agency Technical Report 8350.2. These methods are summarized in sub-paragraphs a through d.

a.   Polar Motion

Polar motion parameters, x_p and y_p, for the dates encompassing the STS-51 mission, were obtained from the U.S. Naval Observatory. The A matrix consists of a rotation about the Y-axis by angle $-x_p$, and a rotation about the X-axis by angle $-y_p$.

The maximum amplitude of these parameters is approximately 0.3 arc seconds. This corresponds to about 10 meters of position difference at Shuttle altitudes.

b. *Sidereal Time*

The B matrix consists of one rotation about the Z-axis by an angle of Λ, where $\Lambda = H_0 + \Delta H + \omega^*(t - \Delta t)$. H_0 is Greenwich Mean Sidereal Time (GMST) at 0^h UT1 on the day of interest. Since 1984, GMST has been defined by Equation 3, where T_u is the number of centuries elapsed since 12^h UT1 on 2000 January 1. The result is in units of seconds of sidereal time, and may be converted to arc on the basis of one revolution per 24 hours of sidereal time.

$$\text{GMST1 of } 0^h \text{ UT1} = 24110.54841 + 8640184.812866 T_u + 0.093104 T_u^2 - 6.2 \times 10^{-6} T_u^3$$

$$\text{(Eq: 3)}$$

ΔH is the Equation of the Equinoxes. It equals $\arctan(\cos\varepsilon \tan\Delta\psi)$, where ε is true obliquity of the ecliptic, and $\Delta\psi$ is nutation in longitude. Both ε and $\Delta\psi$ are computed in the course of generating the C matrix. ω^* is the Earth rotation rate in a precessing reference frame, and is equal to $\omega' + m$, where ω' is the Earth's inertial rotation rate, a constant, and m is equal to $7.086 \times 10^{-12} + 4.3 \times 10^{-15} T_u$. Time t is the time of the state vector being transformed in seconds since the beginning of the day UTC, and Δt is the difference between UT1 and UTC. Values of UT1 minus UTC were obtained from the U.S. Naval Observatory for the dates encompassing the STS-51 mission. These values are kept below 0.7 seconds through introduction of leap seconds into UTC. One second of time corresponds to 15 arc seconds, or about 485 meters of position difference at Shuttle altitudes.

(1) Change in Sidereal Time Matrix. The B matrix is defined as shown in Equation 4.

$$
\begin{bmatrix}
-\omega^* \sin\Lambda & \omega^* \cos\Lambda & 0 \\
-\omega^* \cos\Lambda & -\omega^* \sin\Lambda & 0 \\
0 & 0 & 0
\end{bmatrix}
\qquad \text{(Eq: 4)}
$$

c. Astronomic Nutation

The C matrix consists of a rotation about the X-axis by angle $\bar{\epsilon}$, followed by a rotation about the Z-axis by angle $-\Delta\psi$, followed by a rotation about the X-axis by angle $-\epsilon$. Angle $\bar{\epsilon}$ is mean obliquity of the ecliptic, and is defined by Equation 5, where T is the number of Julian centuries elapsed since fundamental epoch J2000.0 in barycentric dynamical time. The result is in units of arc seconds.

$$
\bar{\epsilon} = 84381.448 - 46.815T - 0.00059T^2 + 0.001813T^3 \qquad \text{(Eq:5)}
$$

$\Delta\psi$ is nutation in longitude, and is defined by Equation 6, where A_i, B_i, a_{1i}, a_{2i}, a_{3i}, a_{4i}, and a_{5i} are constants from the 1980 IAU nutation series, shown in Table 3.222.1 of the Explanatory Supplement to the Astronomical Almanac, and l, l', F, D, and Ω are fundamental arguments of the 1980 IAU theory of nutation. The result of nutation in longitude is in units of 0.0001 arc seconds.

$$
\Delta\psi = \sum_{i=1}^{106} (A_i + B_i T) \sin(a_{1i}l + a_{2i}l' + a_{3i}F + a_{4i}D + a_{5i}\Omega) \qquad \text{(Eq: 6)}
$$

The fundamental arguments are depicted in Equations 7 - 11. The superscript r represents revolutions, and results are in units of arc seconds.

$$
l = 485866.733 + (1325^r + 715922.633)T + 31.31T^2 + 0.064T^3 \qquad \text{(Eq: 7)}
$$

$$
l' = 1287099.804 + (99^r + 1292581.244)T - 0.577T^2 - 0.012T^3 \qquad \text{(Eq: 8)}
$$

$$F = 335778.877 + (1342^r + 295263.137)\,T - 13.257\,T^2 + 0.011\,T^3 \quad \text{(Eq: 9)}$$

$$D = 1072261.307 + (1236^r + 1105601.328)\,T - 6.891\,T^2 + 0.019\,T^3 \quad \text{(Eq:10)}$$

$$\Omega = 450160.28 - (5^r + 482890.539)\,T + 7.455\,T^2 + 0.008\,T^3 \quad \text{(Eq:11)}$$

Angle ε is true obliquity of the ecliptic, and is equal to $\bar{\varepsilon} + \Delta\varepsilon$, where $\Delta\varepsilon$ is nutation in obliquity. $\Delta\varepsilon$ is defined by Equation 12, where C_i and D_i are additional constants from the 1980 IAU nutation series, found in table 3.222.1 of the Explanatory Supplement to the Astronomical Almanac. The result of nutation in obliquity is in units of 0.0001 arc seconds.

$$\Delta\varepsilon = \sum_{i=1}^{106} (C_i + D_i T)\cos(a_{1i}l + a_{2i}l' + a_{3i}F + a_{4i}D + a_{5i}\Omega) \quad \text{(Eq:12)}$$

(1) More Accurate Nutation. $\Delta\psi_c$ and $\Delta\varepsilon_c$ are corrections to be added to the 1980 IAU nutations in longitude and obliquity, $\Delta\psi$ and $\Delta\varepsilon$, respectively. $\Delta\psi_c$ and $\Delta\varepsilon_c$ are defined by Equations 13 and 14, where LS_n, LC_n, OC_n and OS_n are constants from the corrections to IAU 1980 nutation series given in table 3.224.1 of the Explanatory Supplement to the Astronomical Almanac. The results of these nutation corrections are in units of 0.00001 arc seconds.

$$\Delta\psi_c = \sum_{n=1}^{4} (LS_n \sin A_n + LC_n \cos A_n) \quad \text{(Eq:13)}$$

$$\Delta\varepsilon_c = \sum_{n=1}^{4} (OC_n \cos A_n + OS_n \sin A_n) \quad \text{(Eq:14)}$$

26

A_n is equal to $a_n l + b_n l' + c_n F + d_n D + e_n \Omega$, where a_n, b_n, c_n, d_n, and e_n are additional constants from the corrections to the 1980 IAU nutation series given in table 3.224.1 of the Explanatory Supplement to the Astronomical Almanac.

d. General Precession

The P matrix consists of a rotation about the Z-axis by angle $-\zeta$, followed by a rotation about the Y-axis by angle θ, followed by a rotation about the Z-axis by angle $-z$. Angles ζ, θ, and z are defined by the accumulated precession angles adopted by IAU 1976, and are shown in equations 15 - 17, respectively. The results are in units of arc seconds.

$$\zeta = 2306.2181T + 0.30188T^2 + 0.017998T^3 \qquad \text{(Eq:15)}$$

$$z = 2306.2181T + 1.09468T^2 + 0.018203T^3 \qquad \text{(Eq:16)}$$

$$\theta = 2004.3109T - 0.42665T^2 - 0.041833T^3 \qquad \text{(Eq:17)}$$

e. Standard Epoch Conversion

The M matrix is depicted in Equation 18.

$$\begin{bmatrix} 0.9999256791774783 & -0.0111815116768724 & -0.0048590038154553 \\ 0.0111815116959975 & 0.9999374845751042 & -0.0000271625775175 \\ 0.004859003771445 & -0.000027170449221 & 0.9999881946023742 \end{bmatrix} \qquad \text{(Eq:18)}$$

5. Summary

The coordinate conversion process was implemented in C++ (Borland ver. 3.1). The program was run using an IBM compatible personal computer. The computer code is included as Appendix A. The code's accuracy was validated in two ways. First, precession, nutation, and Earth rotation (sidereal time) angles as shown in the Astronomical Almanac were accurately reproduced for a given date. Second, a sample set of transformed TANS GPS coordinates produced by the program was compared with the same set of coordinates, transformed by the Shuttle Navigation group at NASA. Radial position differences were less than six meters (less than 0.18 arc seconds of rotation),

and radial velocity differences were less than 0.007 meters per second. This level of accuracy was considered sufficient, given the differing methods used in making the transformation.

B.   NAVIGATION PERFORMANCE

Following coordinate conversion, the TANS GPS state vectors and Orbiter reference state vectors were input into a second computer program to compute the RSS position and velocity differences. The computer code utilized is included as Appendix B. The results for each data set were plotted, and the mean differences and standard deviations were computed.

Three sets of data were chosen for analysis of the TANS GPS receiver's navigation performance. The principle criterion used in selection of these data points was the absence of any prolonged time interval between navigation fixes for the period under consideration. For the periods chosen, time between fixes is generally 2.5 seconds, with occasional gaps of up to 7.0 seconds. Data recorded during the STS-51 mission was segregated into files labeled A through M, O, P, Q1, Q2, R1, R2, and S through U. In general, the files corresponded to a particular event or activity during the mission. Large files were subdivided into smaller files using a numerical suffix, such as P.001, P.002, etc. The P files corresponded to the crew sleep period between flight day 6 and 7, and contained over half the TANS GPS state vectors collected during the mission. All data analyzed was taken from P files.

Analysis of the results is shown in Table 3, and in Figures 6 through 11. The Figures depict the magnitude of the position and velocity differences, and show the range of error for each measurement, as well as for the reference trajectory. The times immediately preceeding, and immediately following the analyzed data were periods when the receiver was not computing navigation solutions. In general, a fourth satellite had just

Table 3: COMPARISON OF GPS AND REFERENCE TRAJECTORY

File	# of Samples	Elapsed Time m:ss.s	Position Difference			Velocity Difference		
			Mean (m)	Standard Deviation	Figure #	Mean (m/s)	Standard Deviation	Figure #
P.004	98	4:23.5	425.8	83.2	6	1.71	0.76	7
P.005	131	5:41.0	222.5	30.6	8	0.83	0.28	9
P.008	140	6:14.5	322.0	29.0	10	1.60	1.02	11

become usable at the beginning of the period being analyzed, and less than four satellites were usable at the end of the period.

Differences between the position and velocity plots for the different data sets were attributable to a combination of factors. Among these were the number of usable satellites available to the receiver (4, 5, or 6), the corresponding geometry of the usable satellites, and the effects of SA. Of note, SA was implemented on most, but not all satellites in the GPS constellation, causing the accuracy of each measurement to be dependent on which satellites were used for the measurement.

TANSGRAPH, a plotting utility furnished by NASA, was used by the authors to display raw data recorded during the mission. TANSGRAPH plots were generated for files P.004, P.005, and P.008, showing periods when four usable GPS satellites were visible, and the associated PDOP's for those time periods. The plots are included as Appendix C.

In general, the position data was quite smooth, with the TANS GPS and Orbiter reference position error overlapping the majority of the time. Conversely, the velocity data appeared more erratic, with the TANS GPS and Orbiter reference velocities coinciding only a couple of times in one of the data sets. The relatively large velocity errors

29

would introduce significant error if used in a propagation scheme, and would be unsuitable for navigation purposes without some effective form of filtering. A few data points were observed to differ from neighboring points by an anomalously large amount. It is possible that these deviations were produced as an artifact of data recording, or data extraction by the TANSPOST program. NASA has since incorporated a six standard deviation rejection scheme into the GPS processing to eliminate outlying points. (Saunders, 1994, pp. 1-13)
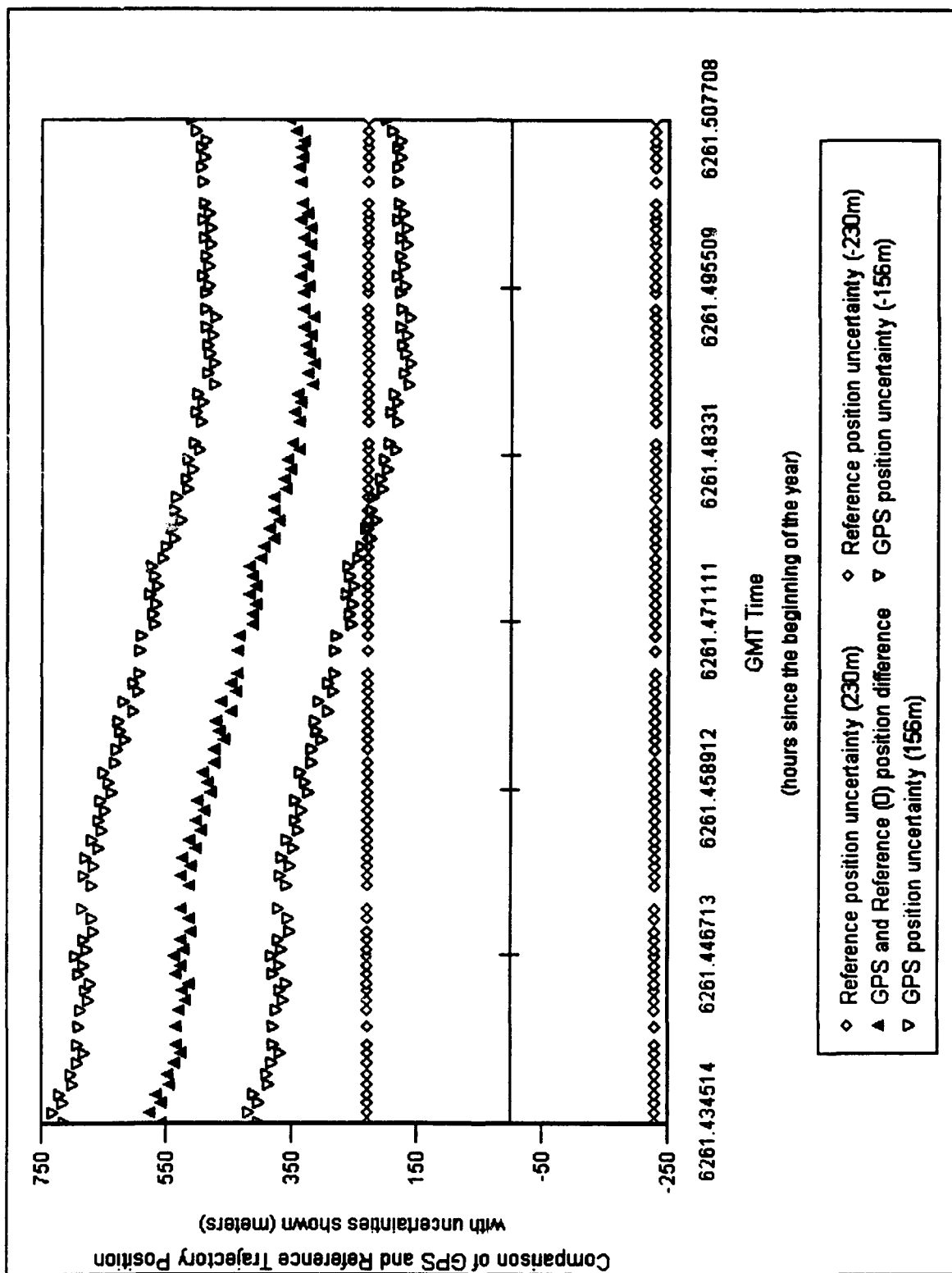
Figure 6: File P.004 Position Differences
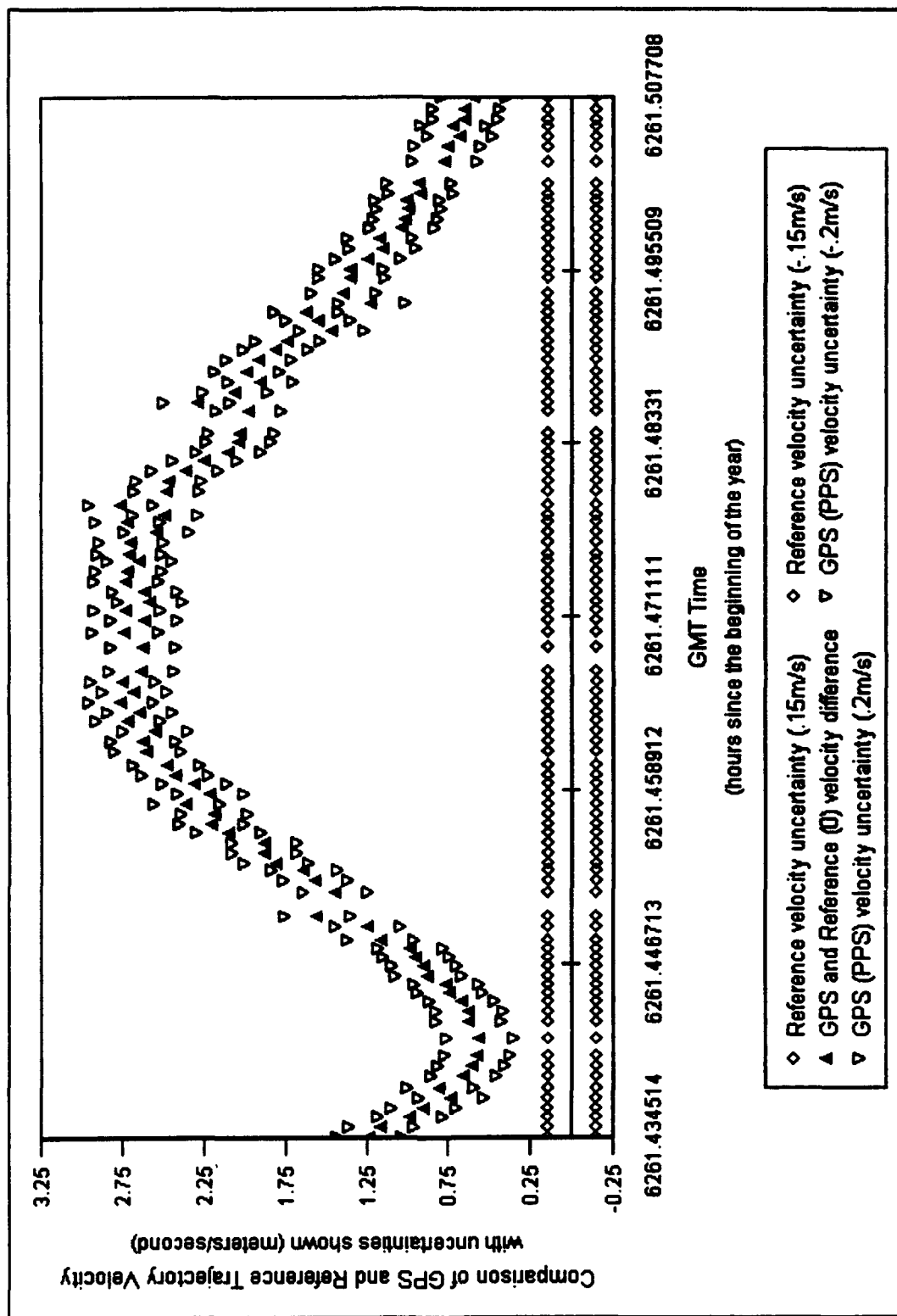
Figure 7: File P.004 Velocity Differences
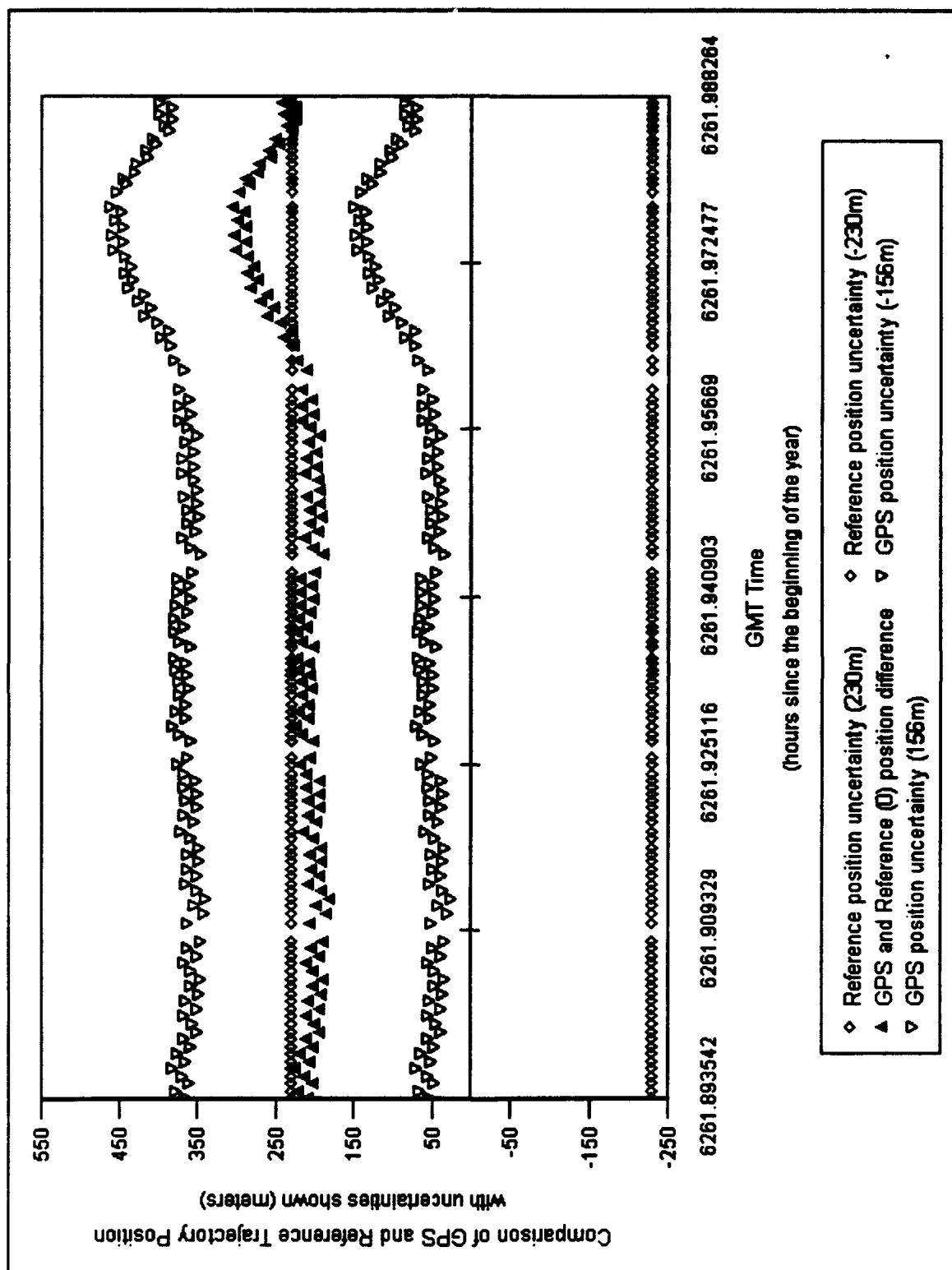
Figure 8: File P.005 Position Differences
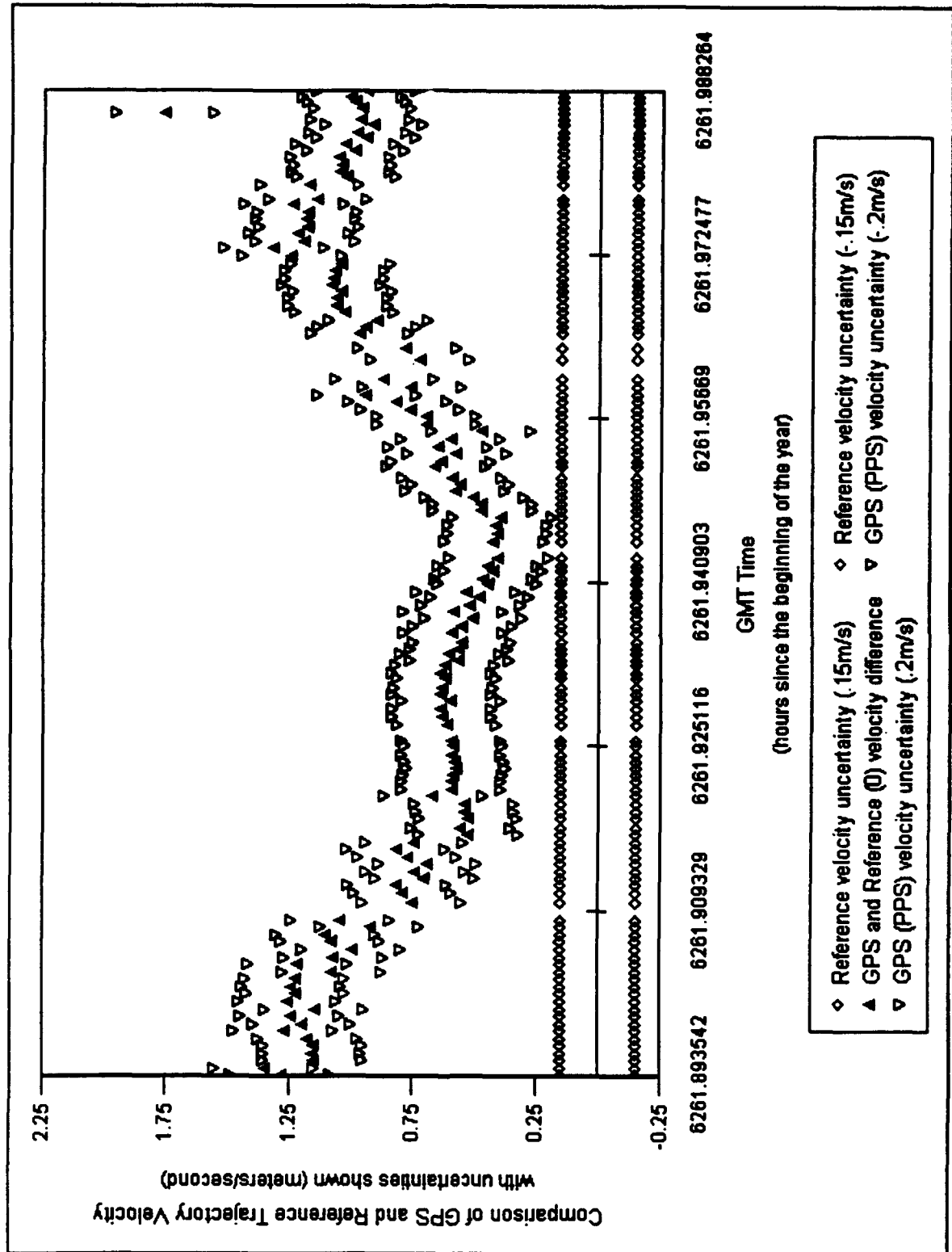
Figure 9: File P.005 Velocity Differences

34
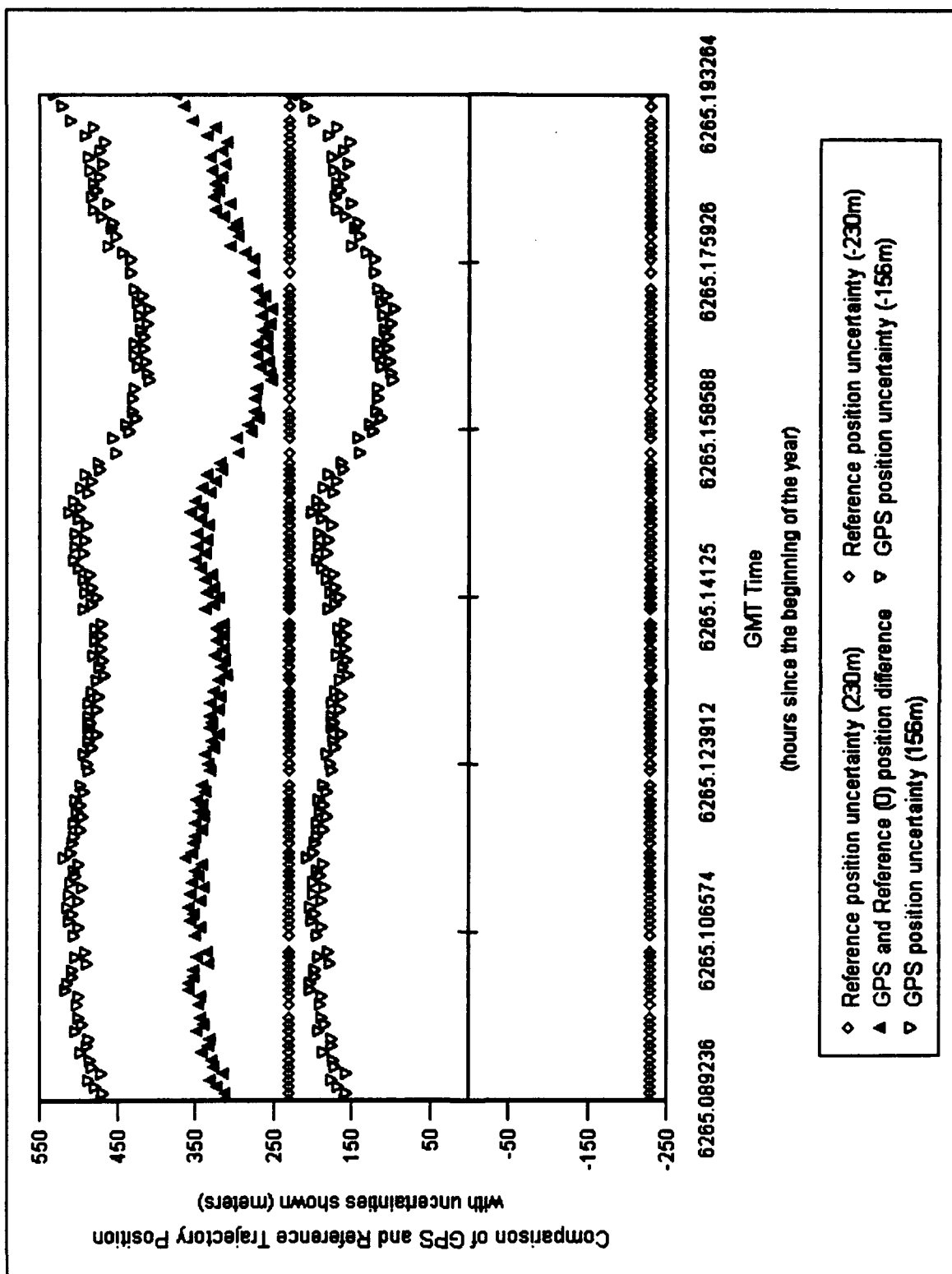
Figure 10: File P.008 Position Differences
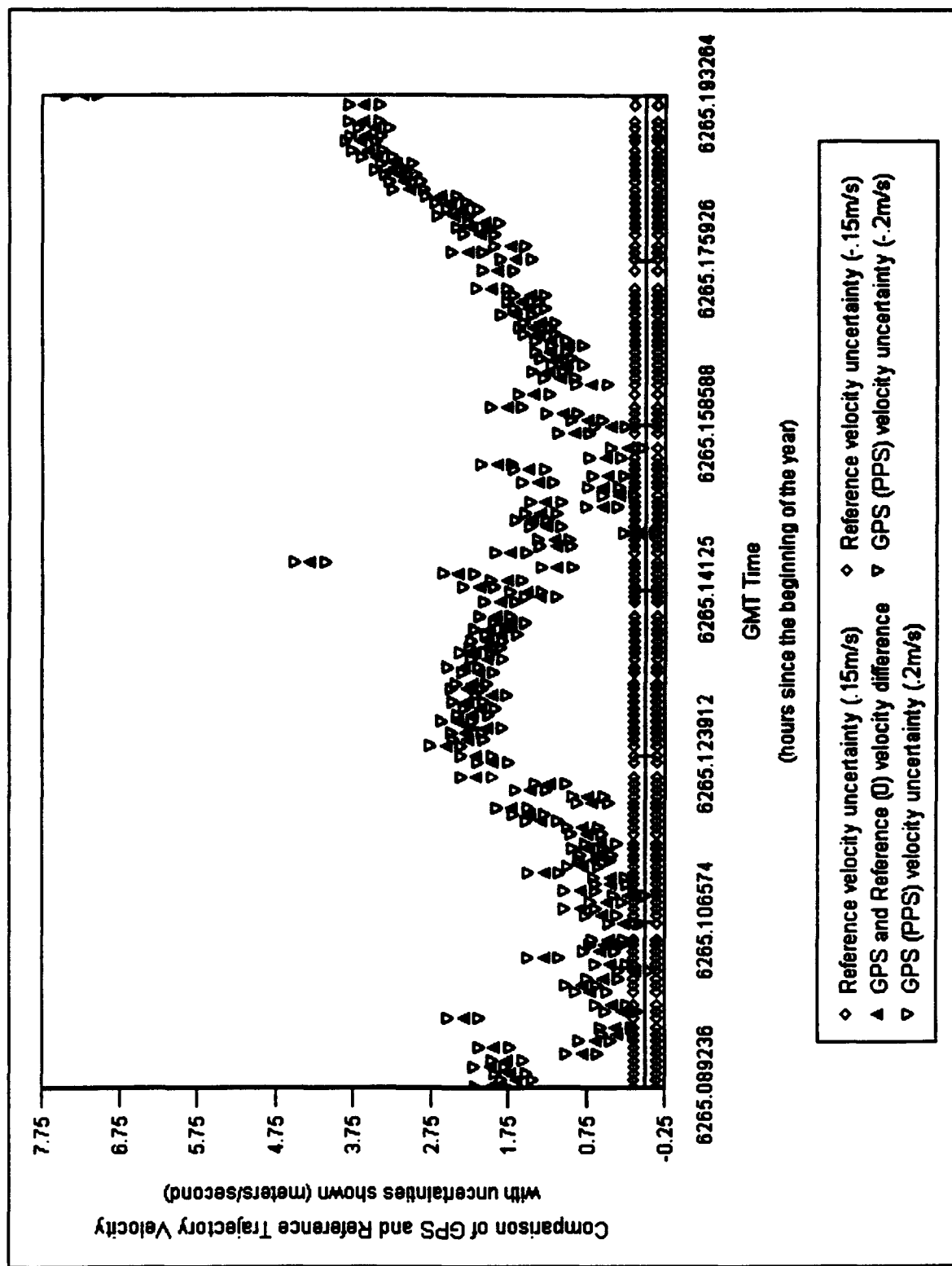
Figure 11: File P.008 Velocity Differences

IV. STATE VECTOR FILTERING

A. INTRODUCTION

1. The Purpose for a Filter

Chapter I discussed how a GPS receiver measures pseudoranges to four satellites in order to solve for a three dimensional position. The GPS receiver calculates the user's position and GPS time by knowing the position of these four satellites from decoding their navigation messages. Pseudorange measurements are made because the GPS receiver can not measure the exact range to each satellite. These measurements are corrupted by ionospheric delays, user clock drift, receiver noise and other errors. Typically a filter is used to characterize some of the noise sources in order to minimize their effects on the navigation solution. A Kalman Filter can be used both within the receiver logic or in a post-processed phase. Additionally, a smoothing algorithm could be applied in the post-processing of data. Most filtering schemes studied today integrate the Kalman Filter with the Inertial Navigation System (INS) by using external informative sources to improve position (LORAN, OMEGA, laser ranging, etc.), velocity (Doppler radar), and altitude (barometric, radar and laser altimeters). In this thesis, a version of the Kalman Filter is implemented and analyzed in the post-processing phase in which user position, velocity and GPS time are known from STS-51's flight. This filter is an adaptation of a Kalman Filter program, written by Dr. Titus, a professor from the Naval Postgraduate School, which uses seven states including position (X, Y, Z), velocity (V_x, V_y, V_z), and time (t). The theory of the Kalman Filter will be described briefly along with the adapted computer code (see Appendix D). The final results will be analyzed to show the effects, both positive and negative, a Kalman Filter has on this post flight data.

B. KALMAN FILTER

1. Theoretical Model

The GPS Kalman Filter is a model of how its state vector is changing in time or how the host vehicle is maneuvering in time. The state vector includes parameters which describe the model, a minimum of which is receiver position (X, Y, Z) and time. The simplest way to describe the Kalman Filter is as a recursive estimator that produces a minimum covariance estimate of the state vector in a least squares sense. The covariance estimate or matrix expresses the statistical uncertainty in the state vector. The uncertainty grows during long periods without measurements. However, when a new measurement becomes available it will be weighed heavily regardless of how noisy it may be unless the filter designer plans accordingly.

Figure 12, from the NAVSTAR GPS User Equipment Introduction guidebook of February 1991, shows a simplified Kalman Filter diagram illustrating how it processes new measurements and propagates in time.
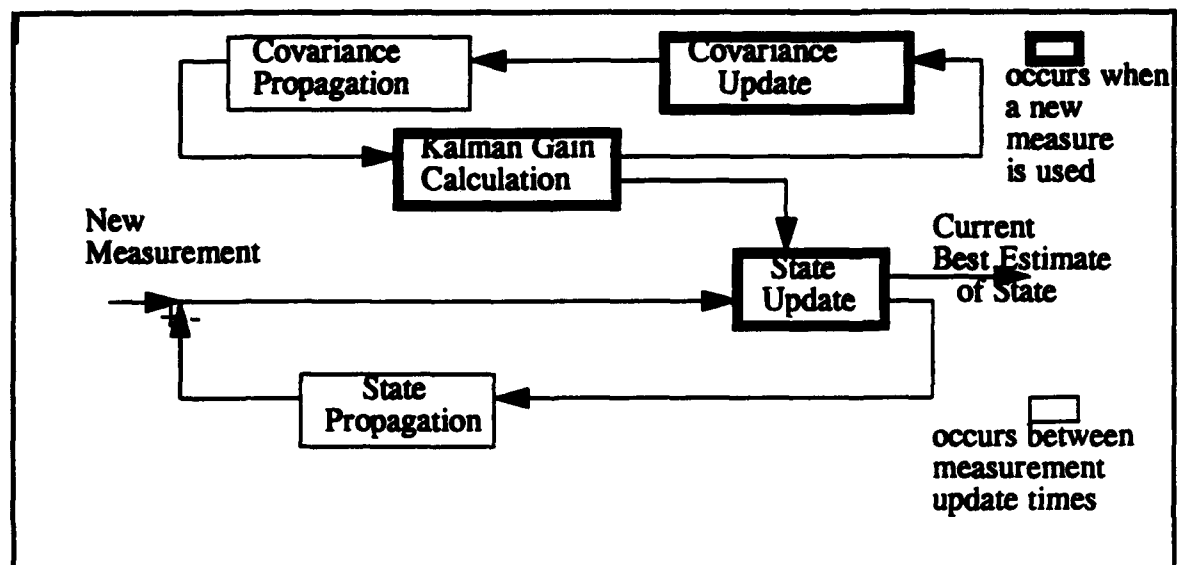
Figure 12: Basic Block Diagram of a Kalman Filter

38

2. Theoretical Equations

The basic process of the discrete Kalman Filter, is to model the state vector, x_t, as it transitions over time, from timestep to timestep. The H_t matrix is called the measurement or observation matrix. The measurement z_t vector is a function of the state given by H_t. The following equations represent two processes, Equation 19 for state and Equation 20 for measurement.

$$x_t = \Phi_{t-1} \cdot x_{t-1} + w_{t-1} \tag{Eq.19}$$

$$z_t = H_t \cdot x_t + v_t \tag{Eq.20}$$

The Φ matrix is the transition matrix between the covariance matrix (P_t) and state (x_t). To account for the uncertainty in the state and measurement models, the noise (Gaussian white) terms w_{t-1} and v_t are included. The Kalman Filter alternates between propagating the state (x_t) and its covariance (P_t) and updating these variables with new measurements. The Q_t matrix is the variance of the state noise and it accounts for the error in the modeling assumptions of Φ. The following equations are used to express the propagation of the state (x_t) and the covariance (P_t).

$$x_t = \Phi_{t-1} \cdot x_{t-1} \tag{Eq.21}$$

$$P_t = \Phi_{t-1} \cdot P_{t-1} \cdot \Phi^T_{t-1} + Q_{t-1} \tag{Eq.22}$$

Updating is defined as incorporating additional measurements into the filter at regular intervals. The state immediately updated is considered to be the most optimal state in the filter. At this point, the new measurement of the state is compared with the propagated estimate. This difference is scaled using the Kalman Gain (K_t) and then used in calculating the new state estimate. Symbols (-) and (+) are used to distinguish between filter estimates immediately before and after a measurement. The following equation expresses the update procedure.

$$x_t(+) = x_t(-) + K_t \cdot [z_t - H_t \cdot x_t(-)] \tag{Eq.23}$$

The next step is to update the covariance matrix (P_t). In Equation 24, the new measurement is weighted by the Kalman Gain (K_t) and differenced with the identity matrix (I). This result determines the degree to which the covariance matrix is improved by the new measurement.

$$P_t (+) = [I - K_t \cdot H_t] \cdot P_t (-) \qquad \text{(Eq.24)}$$

Finally, some understanding of the Kalman Gain (K_t) is required. It is a result figured each time a measurement update occurs. The calculation is not only based upon the propagated covariance matrix of the previous time, P_t (-), but it is also upon the current measurement noise covariance (R_t) and the sensitivity of the measurement to small changes in the state ($H_t = \delta H/\delta x_t$).

$$K_t = P_t (-) \cdot H^T_t \cdot [H_t \cdot P_t (-) \cdot H^T_t + R_t]^{-1} \qquad \text{(Eq.25)}$$

To try to explain this equation, an example from the NAVSTAR GPS User Equipment will be cited. First, assume that the state vector and measurement matrix are both in the same coordinate frame so that the H_t matrix becomes the identity matrix. Second, simplify the notation and matrix formulation to show the Kalman Gain as $K = P / (P + R)$, where P continues to represent the covariance and R represents the measurement noise. So for a large P, or uncertainty in the model, compared to the uncertainty in R, the gain applied to the new measurement is weighed heavily at almost unity. In other words, the propagated state has too large of an uncertainty, so the new measurement is seen as a better estimate and is used as such. For the other case, when there is a large uncertainty in the measurement noise as compared to the state (i.e. $R >> P$), the Kalman Gain is very small and the new questionable measurement is weighted by a small amount.

C. AN ADAPTATION OF THE KALMAN FILTER

1. Analysis of Problem

Examining filtering for the first time, the decisions concerning what to filter and which parameters to characterize were the most challenging aspects to this problem. After researching Kalman Filter theory and obtaining guidance from experts in this field, the problem, at last, became well defined. First it was necessary to study the TANS receiver. It was important to know that Trimble had designed it to work without a filter calculating position and velocity in a deterministic manner. The advantage of accessing this information without previous filtering schemes is that it allows one to create several filtering designs post-flight, having knowledge of the data's behavior. Without access to the original pseudorange and pseudorate information, the problem became one of filtering the output of the receiver using the position (X, Y, Z), velocity (V_x, V_y, V_z) and time (t) parameters. A more elaborate filter might use other parameters, such as, GDOP, Carrier to Noise, and accelerations to assist in weighing new measurements.

Throughout the entire STS-51 flight, there were twenty two periods of TANS GPS operation. Table 4, on the following page, is a modified table from "The First Flight Tests of GPS on the Space Shuttle." It is presented as a reference to show experiment operation periods, Shuttle activity, and related satellite tracking statistics.

After plotting raw position and velocity data, as seen in Chapter III, it was evident that the position solutions tracked smoothly without applying filtering techniques. Choosing one of the P files, GPS51P.005, provided an acceptable base from which to study the Kalman Filter and analyze its advantages and disadvantages. However, to carry the navigation process through potentially long gaps of measurements required a more elaborate filtering scheme not addressed here.

41

Table 4: SUMMARY OF GOOD DTO PERIODS OF OPERATION

Attitude	File(s)	Event/ Activity	Flight Day(s)	Duration	Satellite Tracking >3
-ZLV,-YVV	C,D,U	-	1, 8, 9	> 8 hrs	0%
+YLV,+ZVV	E thru H I thru M	IMU align sleep	3 4,5	>91 mins >8 hrs	0 to 21.9% 0 to 10.7%
+YLV,-ZVV	O P Q1 thru R2	IMU align sleep YLV mnvr	6 6-7 7-8	>2 hrs >9 hrs >9 hrs	0.7% 19.8% 0 to 25.2%
ACTS Deploy	A B	ACTS Deploy	1 1	>2 hrs >11 mins	0% 0%
SPAS Rdvs	S T	SPAS Rendezvous	8 8	>39 mins >3 hrs	0% 0%

2. Diagram of the Kalman Filter

Figure 13 is the schematic of a Kalman filter which uses a signal (z_k) as an input. The MATLAB (Version 4.0) code is included in Appendix D.
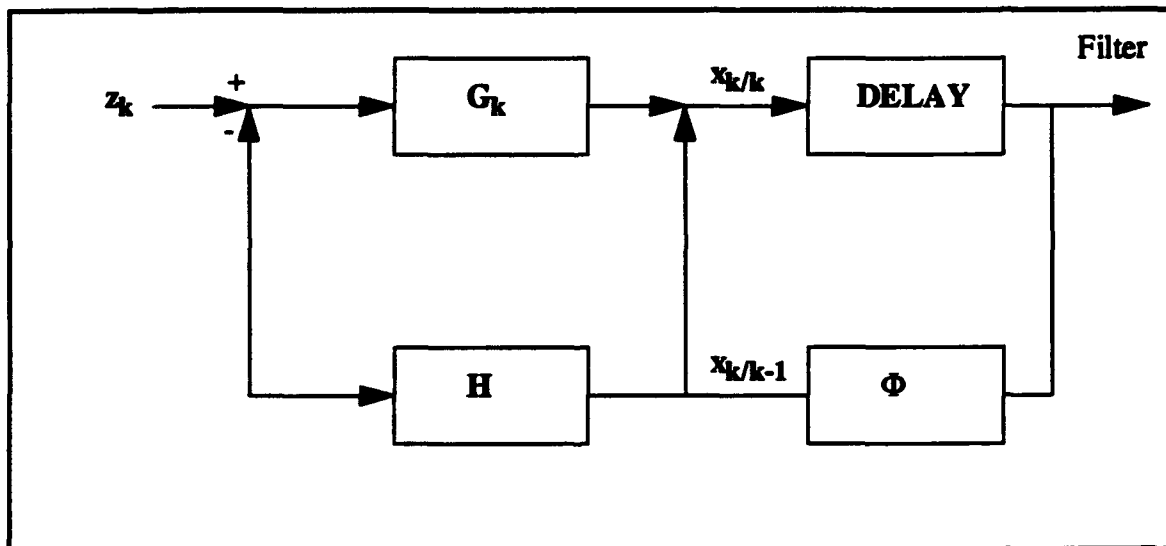
Figure 13: Kalman Filter Design

3. Description of the Kalman Filter Design

The signal input to the filter consists of the current position, velocity, and time. The G_k is a n x m matrix representing the Kalman Gain. H is an m x n measurement or observation matrix which isolates selected states. $P_{k/k}$ and $P_{k/k-1}$ are square matrices [2 x 2] representing the covariance of error of the estimator at k given k observations and at k given k-1 observations, respectively. The R matrix (m x m) is the covariance matrix of the measurement noise and Q (1 x 1) is the covariance of the signal excitation.

The following formulas are used in this Kalman Filter design.

$$G_k = P_{k/k-1} H^T [H P_{k/k-1} H^T + R]^{-1} \tag{Eq.26}$$

$$P_{k/k} = [I - G_k H] P_{k/k-1} \tag{Eq.27}$$

$$P_{k+1/k} = \Phi P_{k/k} \Phi^T + Q \tag{Eq.28}$$

These Kalman Filter equations provide the gains (G) for a typical tracking filter as listed below.

$$x_{k/k} = x_{k/k-1} + G_k [z_k - H x_{k/k-1}] \tag{Eq.29}$$

where

$$x_{k/k-1} = \Phi \cdot x_{k-1/k-1} \tag{Eq.30}$$

$$H = [1 \ 0]$$

$$\Phi = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}$$

$$x_{k/k} = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} x_{k-1/k-1} + \begin{bmatrix} g_1(t) \\ g_2(t) \end{bmatrix} (\sigma_k - x_{k/k-1}) \tag{Eq.31}$$

At the first observation, when k = 1,

$$x_{1/0} = 0, \quad g_1(1) = 1, \quad g_2(1) = 0$$

43

At the second observation, when k =2, then $g_1(2)$ =1 and $g_2(2)$ =1/T (noise free). From the third observation on, the gains will decrease asymptotically to steady state values which depend upon the ratio of the appropriate term of the excitation covariance (Q) and associated measurement noise variance (R).

D.   RESULTS FROM USING A KALMAN FILTER

In the next few pages, plots generated using MATLAB are analyzed to show how well this simple filter performed. Beginning with a look at the behavior of the unfiltered data, it is clear that this series of data illustrates relatively smooth information. Figure 14 and Figure 15 show the Z position and V_z (velocity in the Z direction, termed Z dot) as compared to their estimate, Zhat, generated using the Kalman Filter. The connecting line shows the unfiltered position and velocity data. The position plot, Figure 14, uses the "+" symbol to identify the filtered position estimate. The velocity plot, Figure 15, uses the "o" to discretely show the trend of the filter and one notices the bias in this plot until Q and R are fine tuned. In both of these figures, Q =0.01 and R =1.
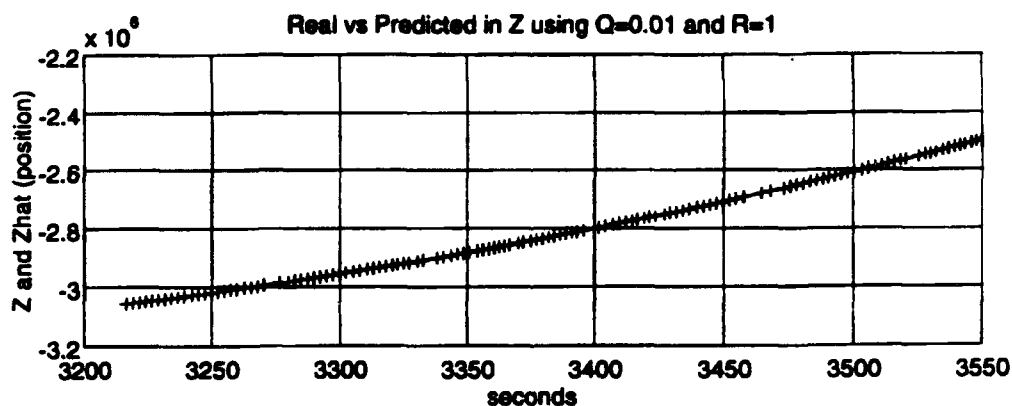
Figure 14: Z Position - Real (-) vs Predicted (+)
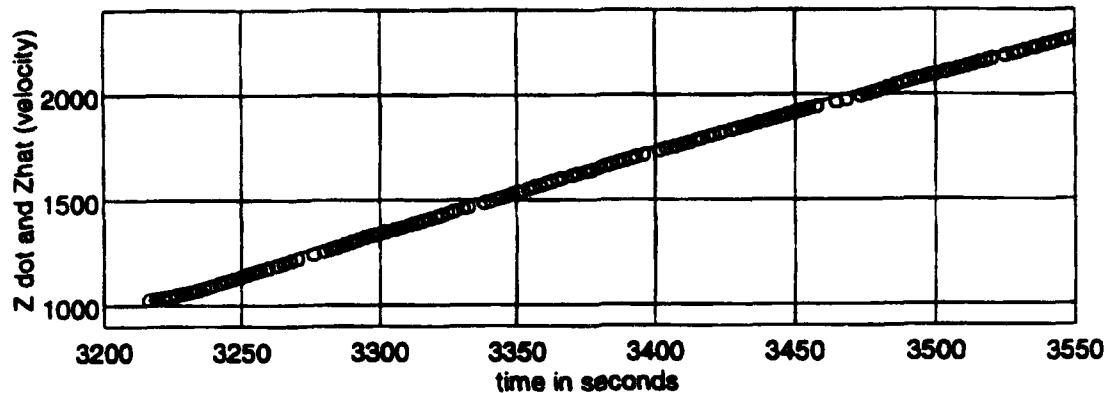
Figure 15: Velocity in Z Direction - Real (-) vs Predicted (o)

In the figures above, one sees that after approximately five inputs, the filter steadies and maintains a track at that level. This characteristic highlights a major issue concerning this filter, initialization. Without initialization, the filter generates a spike. After fine tuning the noise characteristics of **Q** and **R**, the spike in some magnitude remains. Figures 16 and 17 demonstrate the initialization jump for both position and velocity. In order to emphasize the spike, these figures use **Q** =0.01 and **R** =1. In Figure 16, the topmost plot on the following page, the discrete time jumps are plotted against the position gain behavior to show that the time steps are not constant (an average time step of 2.5 seconds with seven seconds being the greatest step) and correlate the changes in time step with the gain changes. With a longer step than 2.5 seconds, one can see the gain increase as it wants to weigh the "overdue" measurement more than its own propagated state.

45
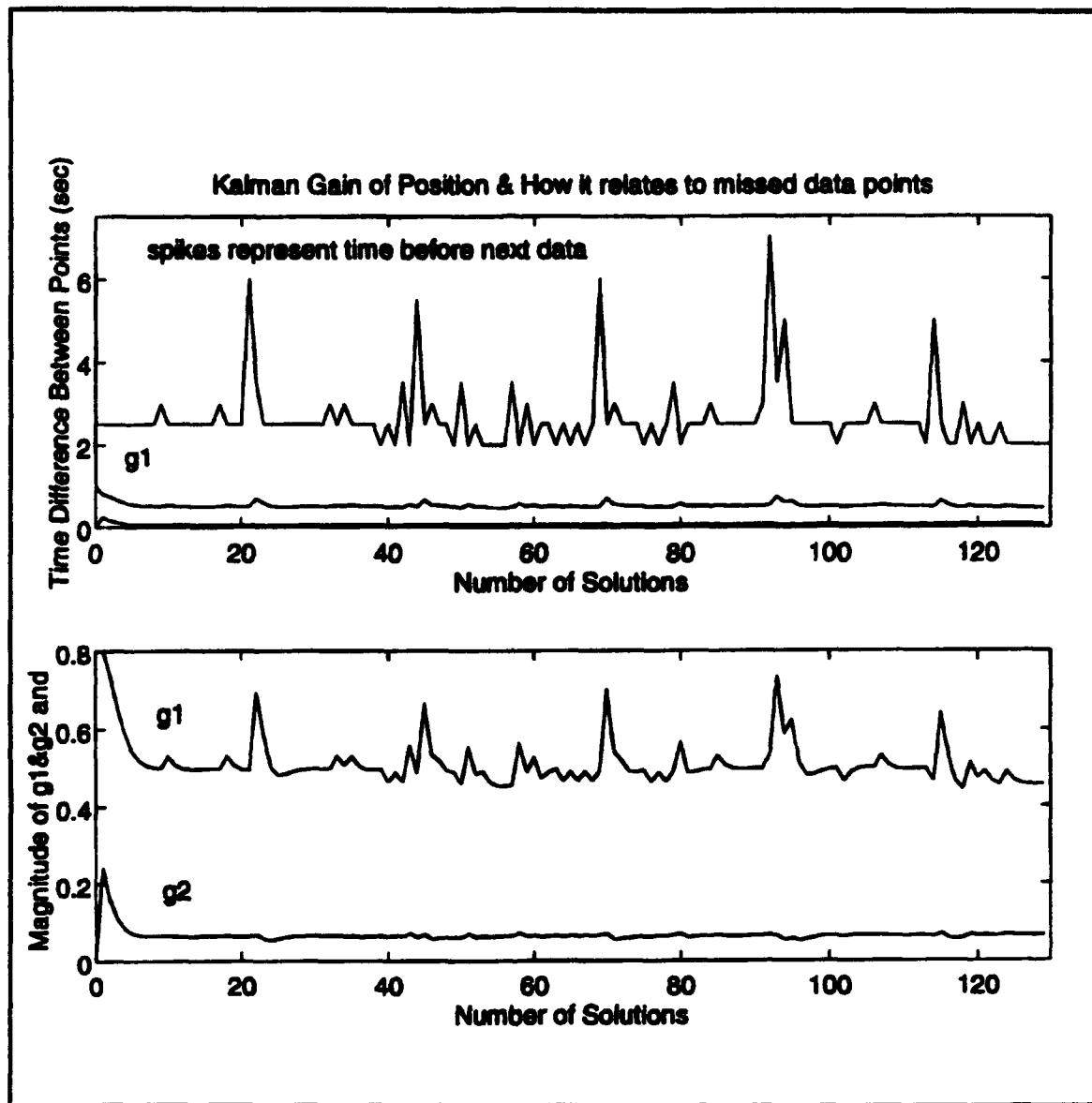
Figure 16: The Position Gain Behavior with Q=0.01 and R=1
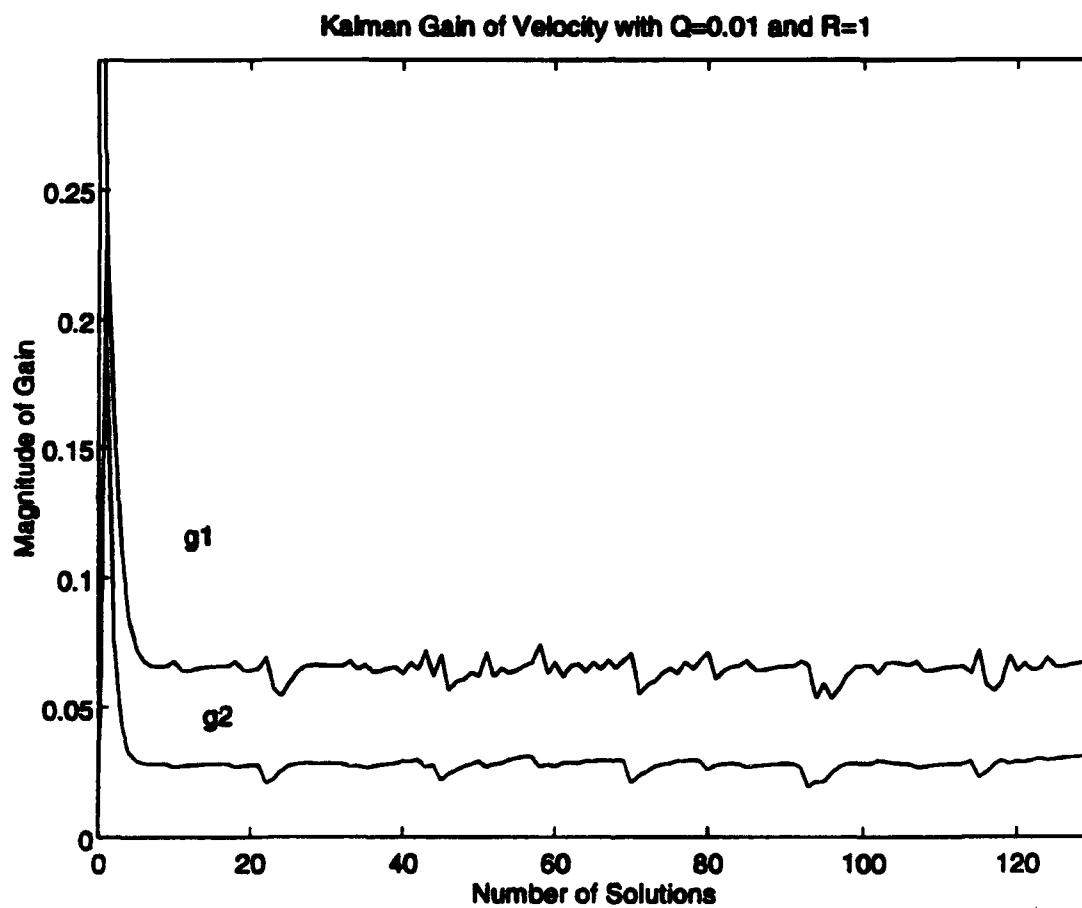
Figure 17: The Velocity Gain Behavior with Q=0.01 and R=1

47

The unfiltered velocity data, in Chapter III, showed a mean of 0.83 meters/second, which is unsuitable for navigation purposes. Figure 18 shows an acceptable filter for position; however, Figure 19 reveals a velocity change which initially jumps up to 25 meters/second and then settles down to ± 5 meters/second state, although stable, it is unacceptable.
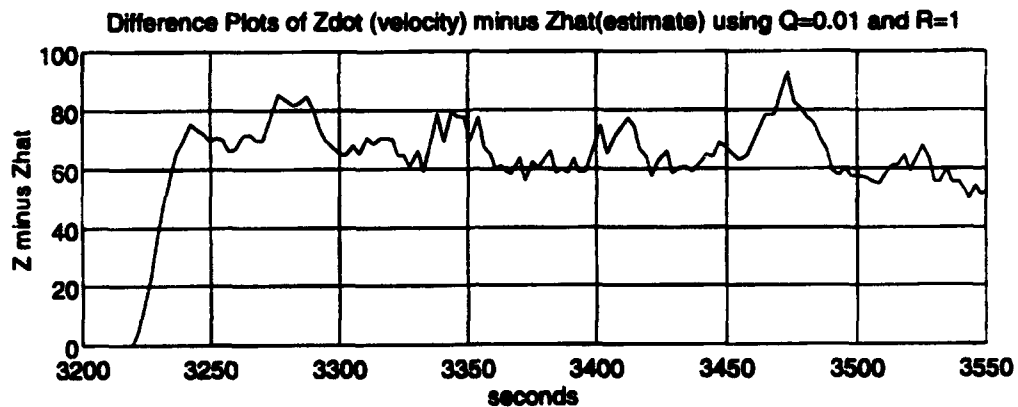
Figure 18: The Difference Plot of Z minus Zhat (the estimate)
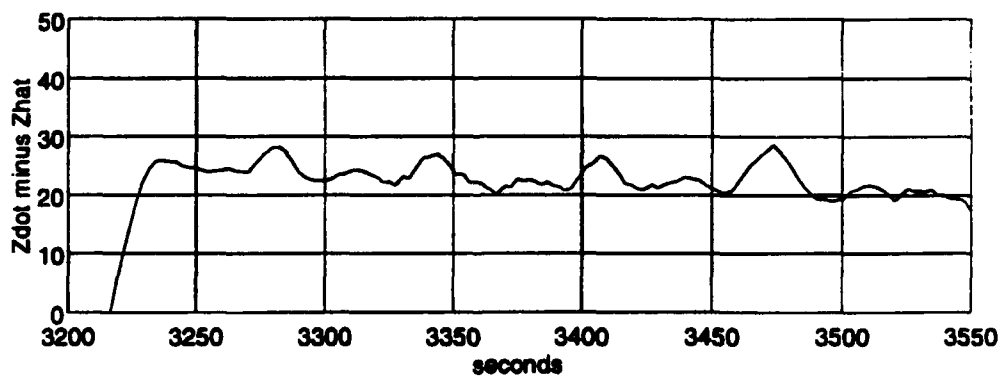
Figure19: The Difference Plot of Zdot minus Zhat (the estimate)

48

If process noise was zero, $Q = 0$, and R was set arbitrarily to one, the estimator stability would converge to zero, as shown in Figure 19. If on the other hand, sensor noise was zero, $R = 0$, and Q was set arbitrarily to one, the filter would take each measurement, without weighing them, and assume they were correct. Figure 20 shows that there is no difference between the input measurement, Z, and the estimate, Zhat. Therefore, the filter cannot be designed to remove the error completely (i.e. Q and R cannot equal zero). It must manage the disturbances, both sensor and process noise. The filter's accuracy depends upon these two noise vectors. (Kaminar, 1993, p.168)
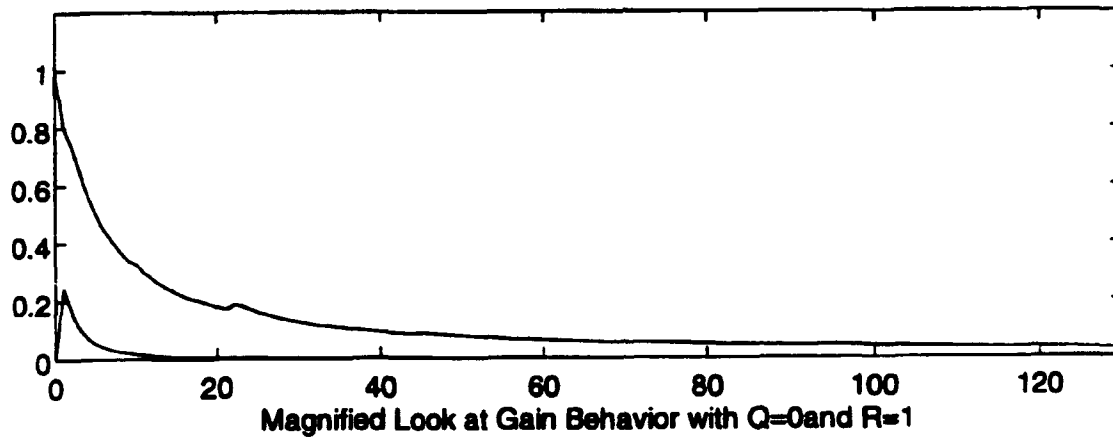
Figure 19: The Position Gain Behavior with Q=0 and R=1

49

Real vs Predicted in Zdot using Q=1 and R=0
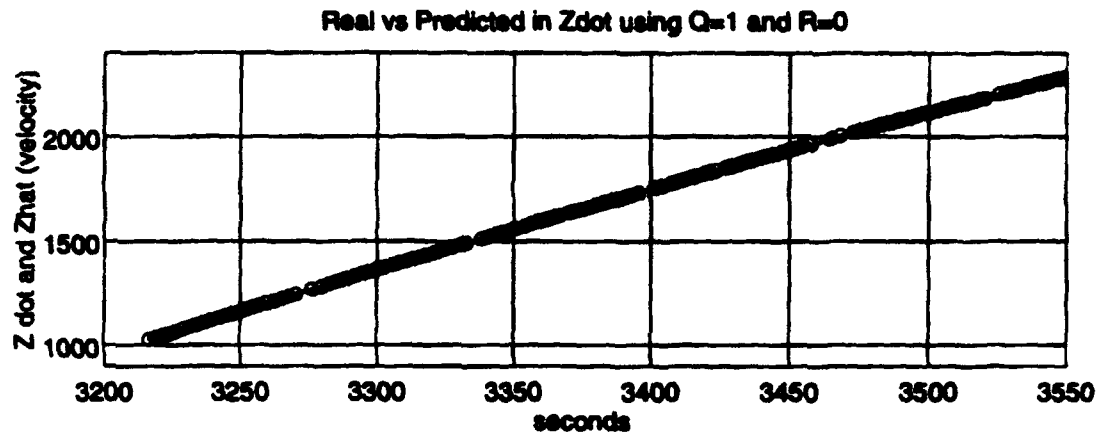
Figure 21: Velocity in Z Direction - Real (-) vs Predicted (o) with Q=1 and R=0

The gain, simply, is a weight emphasizing the incoming measurement. If the measurement is noisy, it should be de-emphasized by the filter. This is accomplished by increasing the sensor noise (R). However, in this case, the information in the file studied appeared to be steady which would suggest a low R. As for the process noise (Q), its value represents the errors in the modeling assumptions.

If only worried about filtering position, a Q of 0.01 proves to be a viable setting. But in order to filter the velocity, a higher Q became necessary. After running many combinations of Q and R, the filter began to provide acceptable velocity variances. The following figures illustrate the differences between conditions of Q and R as they are changed. Figure 22 shows the filtered velocity differences using Q = 1 and R = 1 which can be compared to Figure 23 in which Q = 1 and R = 0.05, and finally, the previous two plots can be compared to the preferred setting in Figure 24, Q = 3.5 and R = 0.05.

One still recognizes the spike, albeit its effect is considerably reduced in Figure 24. This will only be corrected with an initialization technique not addressed in this thesis.
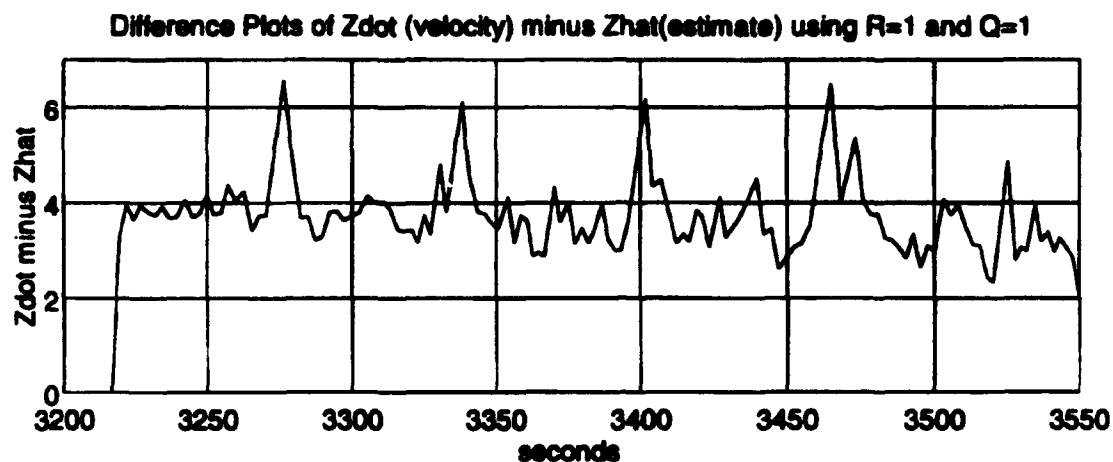
Figure 22: The Difference Plot of Zdot minus Zhat (Q=1, R=1)
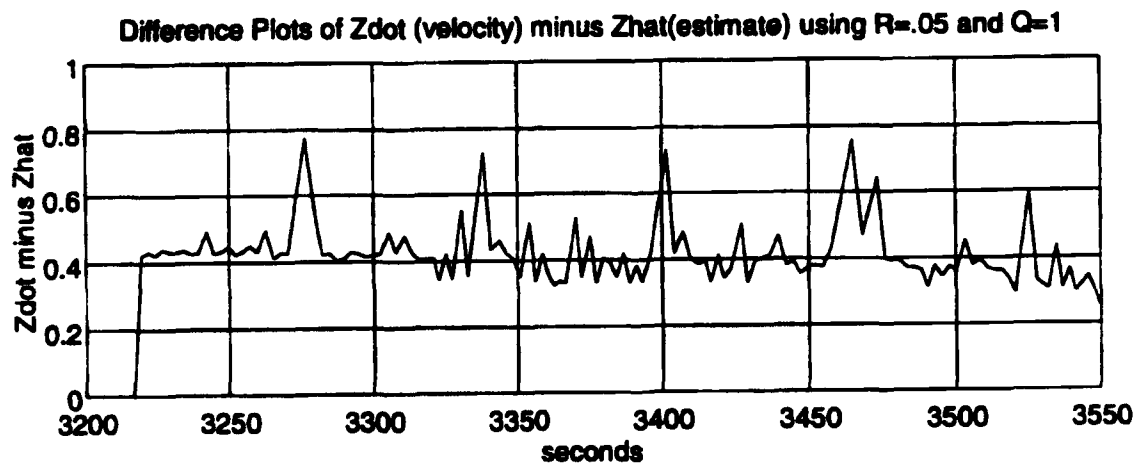
Figure 23: The Difference Plot of Zdot minus Zhat (Q=1, R=0.05)

Difference Plots of Zdot (velocity) minus Zhat(estimate) using R=.05 and Q=3.5
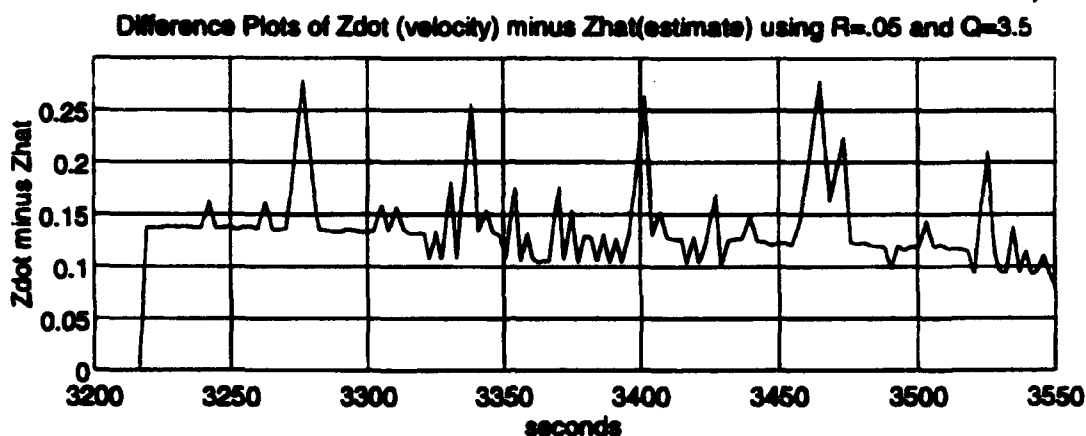
Figure 24: The Difference Plot of Zdot minus Zhat (Q=3.5, R=0.05)

E. SUMMARY

The Kalman Filter has many applications and derivations to suit ones navigation requirements. The advantage to implementing this filter in software after the GPS receiver has processed the pseudorange and pseudorate information, is that it is both an inexpensive approach to experimental testing and it allows access to raw data, untouched by an unproven filter. Without any previous Shuttle GPS data to characterize, NASA would be operating in unchartered territory. Therefore, it was beneficial to start at the lowest level, the experimental stage, before implementing the futuristic triple redundant GPS navigation system. The disadvantages to filtering are basically the same as the advantages listed above, but for different reasons. This relatively inexpensive SPS receiver, if replaced by a PPS version with a filtering scheme using other characteristics (GDOP, etc.) to weigh incoming measurements, would have resulted in an entirely different thesis. In addition to the increased cost, the filter would have returned processed and not raw data, unfortunately, preventing some types of post-flight analysis.

V. ERROR SOURCES

A. PREDOMINANT ERROR SOURCES

In Chapters II and IV, several errors were addressed briefly; however, in this chapter an exhaustive list will be discussed. The major error in the navigation solution is generated from the DOD's use of Selective Availability (SA). System accuracy may be degraded by DOD, to fixed levels by dithering (a method of moving the locations of the bits about in time using a technique "authorized users" may remove). The current policy implements SA at a level giving a 100 meter horizontal (two standard deviation) error. This intentional error is the largest one and has the greatest impact. Although other errors are addressed, unless a method such as differential or relative GPS is incorporated, or a PPS receiver is utilized, these smaller errors control less than one half the magnitude of the total error. The additional errors which will be addressed are atmospheric delays, clock differences, ephemeris error, multipath, receiver noise, and Dilution of Precision (DOP).

Atmospheric delay is caused by the ionosphere and troposphere. The ionosphere is a layer of free electrons and ions above the atmosphere from approximately 100 km to 1000 km. Pseudoranges are significantly lengthened because these charged particles slow transmissions from the GPS satellites. The amount of distortion is directly proportional to the number of electrons along the transmission path. The error can range from 5 to 40 meters, depending on the scenario. For example, a low elevation angle to the satellite affects the signal more than an overhead view.

Four additional factors affect the electron concentration in the ionosphere. They are the solar cycle, time of year, time of day, and latitude. The sun follows an 11 year cycle with the next peak predicted to occur around the year 2001 to 2002 which will

53

have its maximum influence on the ionosphere. Spring equinox brings the greatest seasonal levels of electron concentration. The most active time of the day is at 1400 local time as compared to night when the ionosphere causes its least interference. To eliminate errors due to the ionosphere, the preferred method is to use both the L1 and L2 carrier frequencies, since its effect varies with frequency. Use of both frequencies depends upon the receivers capabilities. The TANS, for example, uses only one. However, in lieu of access to both, modeling the ionosphere is another viable option.

The troposphere, represented by the molecules in the lower atmosphere, also lengthens the transmission path. The tropospheric effect is proportional to the number of molecules above it, which are separated into categories of water and everything else. The everything else division is termed the "dry" category. It can be modeled as a function of elevation angle. The water, or "wet" component is more difficult to model. The errors from these tropospheric effects range from 0.15 to 2.5 meters. Their effect is not frequency dependent; therefore, it is necessary to use a model to decrease the average error of approximately 2 meters.

Errors caused by clock bias and clock drift seem unavoidable. Sources include the space segment (i.e. the GPS satellite clock errors), the control segment (i.e. the ground station clock errors), and the user segment (i.e. the receiver's clock errors). Using four satellites to arrive at a three dimensional solution, the user clock bias is determined as one of the four unknowns; therefore, it is not included in this error budget. The paper written by R.J. Milliken and C.J. Zoller entitled, "Principle of Operation of NAVSTAR and System Characteristics," states that "individual space vehicle clocks, although highly stable, may deviate as much as 976 microseconds from GPS system time." The receiver typically employs the clock correction coefficients (available in the navigation message) in order to correct this offset. The ephemeris errors and the control segment clock errors have a similar problem; therefore, they can be addressed together. Briefly,

54

each of the GPS satellites transmit their respective ephemerides. This information is updated by the Master Control Segment based on monitoring the individual space vehicle navigation signals by four ground stations. This method is a type of inverted range process. The process, albeit a relatively accurate one, still has residual uncertainties. Therefore, the combined effects in both space vehicle clock offsets and ephemeris determinations are estimated to be approximately 1.5 meters. (Milliken and Zoller,1980,p. 9)

A signal is not restricted to following a direct path, but may bounce off electromagnetically reflective objects. Multipath results from having more than one propagation path from which the range measurements are made. This error source is more likely to occur at low elevation angles. The GPS system is designed to minimize the effects of multipath by using L Band which tends to diffuse, as a signal, rather than to reflect. In addition, GPS receivers are typically designed to reject multipath by setting signal noise level thresholds above the interfering signals and by limiting the elevation angle setting to greater than five degrees. Ultimately, multipath error is estimated to be 1.2 meters.

The error attributed to any receiver depends upon its original design, construction, and logic. This effect is termed receiver noise and resolution. The hardware and software chosen for the individual receiver determines its level of noise generation. The noise can be generated from thermal interference, quantization inaccuracies, and dynamic lag. The latter is due to the host vehicle's level of maneuvering. The estimate of error in this category ranges from 1.5 meters to 7.5 meters.

In order to determine the magnitude of the user position errors in the GPS navigation fix, one needs to combine the geometry of the four chosen satellites and the magnitude of the ranging errors. The Geometric Dilution Of Precision (GDOP) parameterizes the satellite geometry using PDOP (three dimensional position) and TDOP. For the purposes of this experiment, the PDOP setting is the focus. Typical settings for PDOP are

six or less. However, in this experiment, this parameter was allowed to be an unhealthy 20 in order to lock onto four satellites as often as possible. It is difficult to determine how much error this setting induced considering values greater than six are not written about in literature. Using lower PDOPs of about 2.6, produce error estimates that range from 5.8 to 10.1 meters. Therefore, the only undisputable conclusion is that lower PDOPs are better.

B.   ERRORS SPECIFIC TO EXPERIMENT

Before this GPS On-Orbit Demonstration (GOOD) could be used by the STS-51 flight crew, potential errors inherent to this experiment had to be addressed by NASA engineers. The problem areas looked at and studied before the flight were the effects from the Shuttle windows, foam spacing dimensions, viewing angle restrictions, cockpit noise interference, and optimal antenna placement with respect to Shuttle attitude.

One of the many studies performed, prior to this experiment, looked at the Shuttle's window layout and the possible effects cause from the layers of panes and the varied dimensions. Figure 25 shows the locations of the eight windows, along with their numbering scheme. These window numbers were used to identify and record placement of the antennas during the flight.
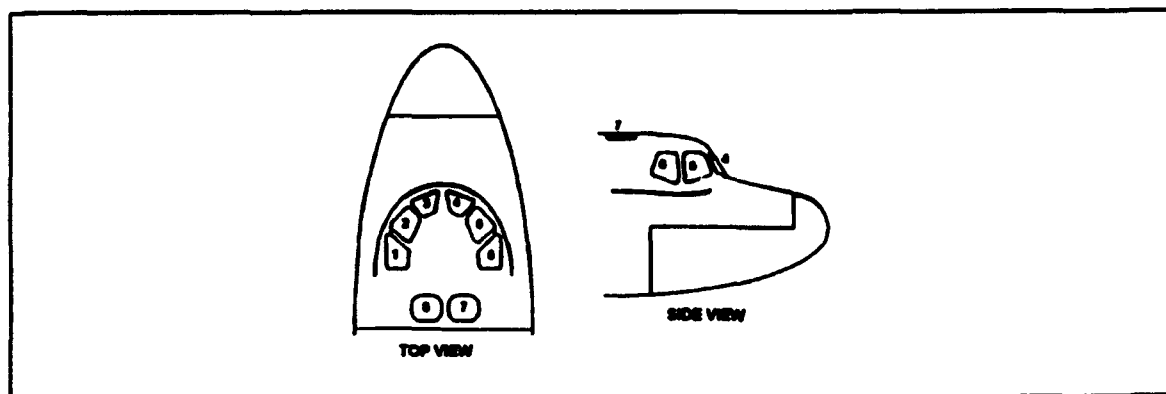
Figure 25: Orbiter Window Locations

56

Each of the eight windows has three panes. These three panes are separated by two gaps varying in distance depending on which window is addressed. The windows are similar in pairs. For example, windows one and six have the same dimensions and so on. The spacing between the outer and middle pane varies because the crew module floats inside the forward fuselage of the Orbiter. Additionally, the size and shape of the windows vary. The lengths of the windows range from 25.9 to 48.9 inches and the widths vary between 14.2 and 34.0 inches. Figure 26 shows the dimensions for all of the windows.
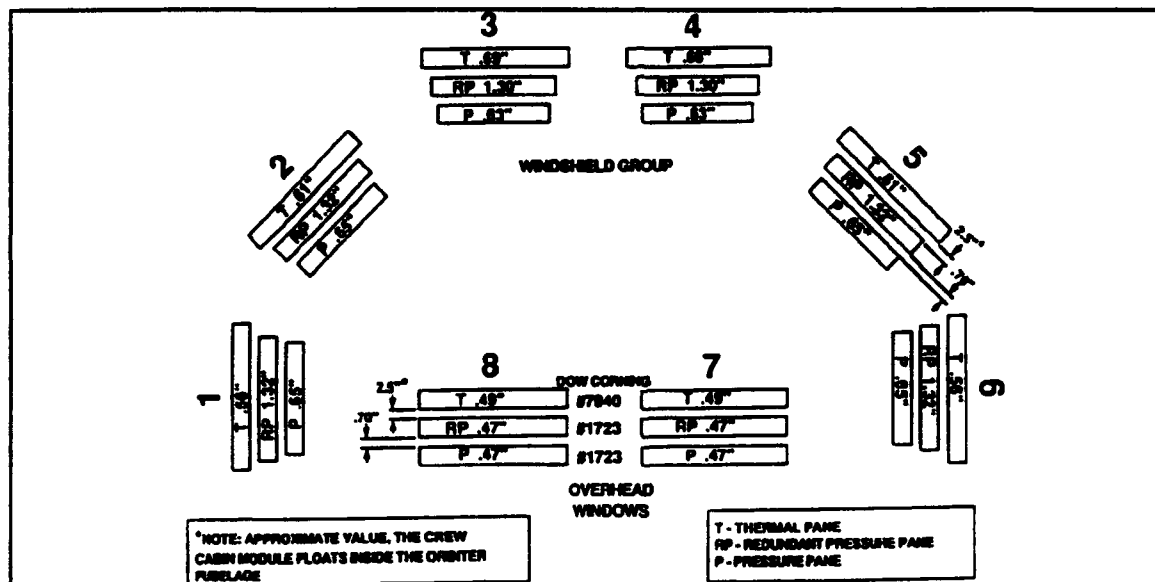
Figure 26: Window Pane Configuration, Thickness and Size

These characteristics are important because in order to study the antenna interaction with the windows, the environment needs to be fully understood. In addition to the multi-pane problem, the antennas were not allowed to be placed directly against the Orbiter's window panes (to avoid damaging the ultraviolet radiation protection coating). The combination of both unknowns required a study of the field of view parameters, sig-

57

nal strength and frequency shifts. An actual mock-up was not available for studying the full effects that the window panes imposed on antenna performance. Instead the tests were simulated as accurately as possible using samples of a thermal pane and two pressure panes together outdoors at NASA's antenna range. As Table 5 shows, using three panes in a simulated fashion, the best reflected power percentage and frequency shift behavior occurred using three inches of spacing. The trade off is in field of view. The optimum field of view happens when the antenna is pressed against the window providing a maximum range of 119° to 153°. Due to the effects listed below and knowing the antenna can not be placed at a zero spacing, the worst case using the three inch spacers is identified at a reduced range of 97° to 140°.

Table 5: EFFECTS OF WINDOW GLASS ON ANTENNA

Number of Panes	Spacing between antenna and panes (inches)	Reflected Power (%)	Frequency Shift (MHz)
0	N/A	0.01	0
3	0	79	172
3	0.5	45	89
3	1	5	0
3	2	0.04	0
3	3	0.01	0

The tests results reported above provided NASA with a qualitative assessment of possible behavior which is attributable to the simulation restrictions in this study. As a final check, the GPS antennas were placed inside *Discovery* while sitting on the launch pad at Kennedy Space Center. The TANS unit was able to track GPS satellites even with considerable blockage of view due to the Orbiter's external fuel tank and the launch tower.

Unfortunately, position errors were difficult to characterize due to large multipath effects from surrounding bodies. Although the three inch spacers gave higher signal levels, STS-51 decided to use the one inch spacers for their flight in order to increase the field of view.

An unsuccessful attempt to fly a similar version of this experiment aboard STS-56 demonstrated a "false-lock" phenomenon. This condition occurred when the antenna/preamplifier units were placed near a PGSC. After thorough research, NASA discovered that the Grid laptop computers were emitting an interfering signal at or near the GPS L1 frequency. It was also discovered that this interference was magnified by metal enclosures, such as the Orbiter's crew cabin. An RF absorber was designed for this particular reason, to counter these interferences inside the crew cabin. A maximum size for these RF absorbers was derived from window viewing requirements. The only other option available to minimize electromagnetic interference was by setting the receiver's signal threshold above the PGSC's noise level. The following figure, Figure 27, shows that although the signal threshold of 5 AMUs was set, the TANS receiver continued to lock on and track the computers.
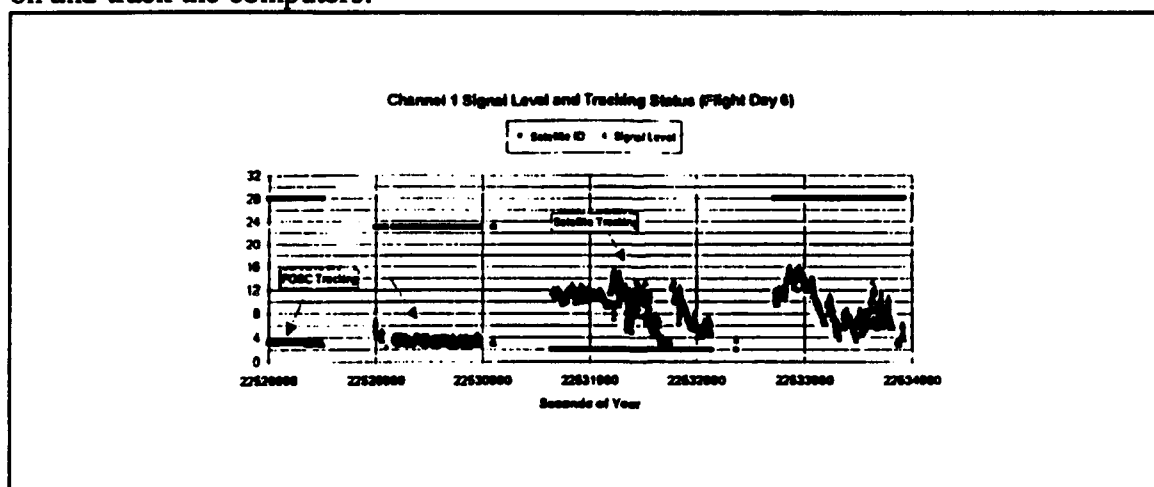
Figure 27: PGSC Interference Signal

59

Additionally, Trimble Navigation modified their signal processing software; however, it was unavailable for this flight. In Figure 27, one can see the Doppler behavior difference between a "false-lock" on the PGSC and the true tracking of a GPS satellite.

C. POST FLIGHT ANALYSIS

The Shuttle operated in three different attitudes during this experiment. These are shown in Figure 28. Of the three attitudes, the worst condition was Attitude 3. The antennas, all facing toward the earth, were never able to get four satellites in view in order to calculate a three dimensional solution. In fact, the best attitude of the three tested was Attitude 1. The "P" series of files were obtained during a sleep period in Attitude 1 between flight day six and seven and were used extensively in post flight analysis. For this period, the three antennas were placed in windows seven, six, and four. Out of approximately 2500 samples, 1464 were found in the "P" series of files.
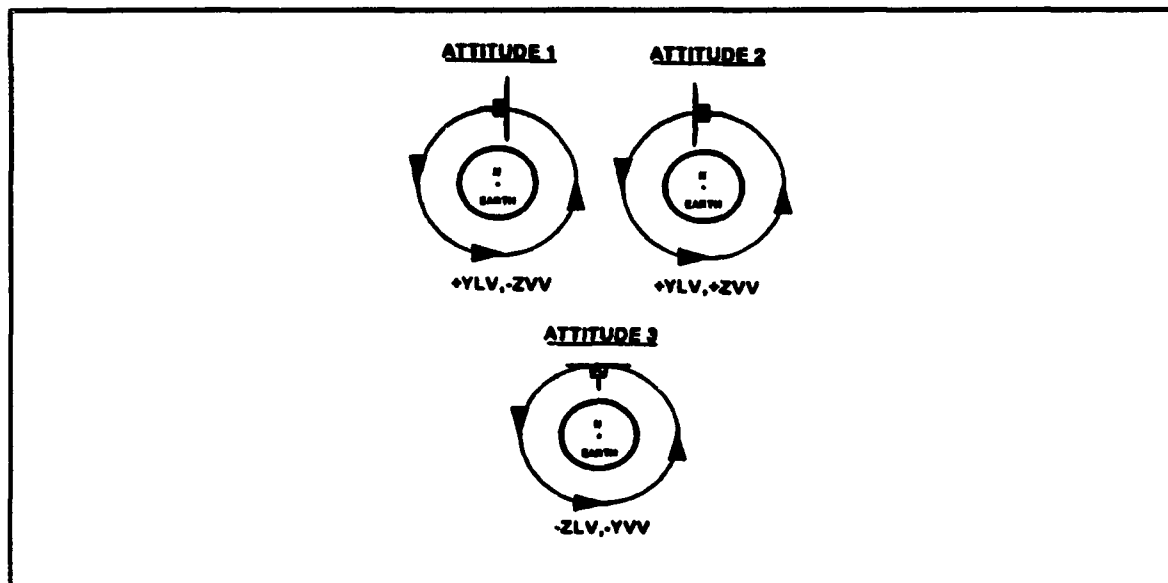
Figure 28: Orbiter Attitudes During GOOD DTO Operations

In conclusion, the attitude of the Orbiter provided a suboptimal environment for viewing the GPS constellation given that the antennas were required to be inside the crew cabin. Placing the antennas outside in the Orbiter's Payload Bay, for example, would improve the performance of the receiver because the issues concerning foam spacing and cockpit noise would disappear. However, the field of view and other potential errors would need to be characterized. Finally, the errors discussed in this chapter were recognized by NASA and future follow-on experiments will be modified to minimize these problems areas.

VI. CONCLUSIONS

A. FLIGHT HARDWARE

One of the chief aims of the GOOD DTO was to keep costs to a minimum. This goal was readily achieved, but at the expense of some capabilities that might otherwise have been available. Two specific improvements in flight hardware would greatly enhance the performance of GPS aboard the Space Shuttle. Each of these improvements would, however, have a significant cost associated with it.

The first suggested improvement is the use of space qualified equipment. This feature would permit components to be located outside the friendly environment of the crew cabin, in the Shuttle's payload bay. This is particularly desirable in the case of the antennas, whose in-cabin field of view is greatly restricted. Better visibility should enable four satellites to be in view for a much higher percentage of time, resulting in more consistent availability of the GPS state. A corresponding reduction in PDOP would also be expected, correlating to improved GPS navigation accuracy.

The second suggested improvement is the use of a Precise Positioning Service receiver. This is particularly desired because of the unsatisfactory velocity results obtained by this experiment due to Selective Availability (SA) . The velocity accuracy of the PPS is 0.2 meters per second. This is the order of magnitude of the desired accuracy for velocity information. The velocity accuracy of the Standard Positioning Service is classified, but errors observed during this experiment were on the order of 2 meters per second. Velocity errors this large are unsuitable for use in navigation, as even small velocity errors cause large position errors when used for propagating states. Use of a Kalman Filter earlier in the receiver's logic would smooth both position and velocity to an acceptable level. While a filtering scheme may succeed in reducing some of the error,

62

the only way to consistently achieve desired velocity accuracies is through use of a PPS receiver.

B.   FLIGHT SOFTWARE

The flight software functioned as it was designed, successfully recording data for postflight analysis. The only desirable modification to the software would be the addition of a relatively fast filtering algorithm, such as a version of the Kalman Filter designed to smooth GPS state vector output. Again, incorporation of this feature would carry an associated cost of further integration and testing.

C.   FUTURE APPLICATIONS

Each of the suggested improvements to the GOOD DTO are already incorporated as part of the design of the GPS navigation system being designed for the Shuttle. (Kachmar, 1993, pp. 313-326). No plans exist to fly the GOOD DTO, as configured on STS-51, again at a future date. However, portions of the experiment have been utilized for two Shuttle missions since the September 1993 STS-51 flight, and STS-66 is scheduled to carry a portable GPS receiver, with antennas mounted in the payload bay, later this year. The GOOD DTO has shown that good on orbit position accuracy can be obtained using an inexpensive portable GPS receiver. The output of such a receiver can be used by any application or experiment requiring precise timing or position information.

APPENDIX A

```
/*******************************************************************/
/*                                                                */
/* This program transforms a state vector from the ECEF_WGS_84 to the*/
/* ECI_M50 cartesian reference frame.                             */
/*                                                                */
/* Author: LT Stephen P. Rehwald, USN                             */
/* Date:   06 March 1994                                          */
/*                                                                */
/* Functions from "Numerical Recipes in C", Press, W.H., et al,   */
/* Cambridge University Press, 1988 were utilized in this program. */
/*                                                                */
/* (As written, this code is valid for state vectors having time  */
/* elements between MJD 49247.0 and MJD 49248.0, as this fulfilled the */
/* author's needs. Other times may be used by incorporating       */
/* corresponding IERS polar motion, UT1_UTC_OFFSET, and applicable leap */
/* second adjustments. Slight modification of the routines for    */
/* computing UT1 time and for interpolating polar motion data will be */
/* required to reflect the newly incorporated data.)              */
/*                                                                */
/*******************************************************************/


#define PI 3.14159265358979932385
#define DEG_TO_RAD (PI/180.0)
#define ARCSEC_TO_RAD (PI/(180.0*3600.0))
#define SEC_TO_RAD ((2.0*PI)/86400.0)
#define ONE_REV 1296000.0
#define DAY 86400.0
#define CENTURY 36525.0
#define OMEGA_PRIME .000072921151467


/*******************************************************************/
/* NUMBER_OF_VECTORS is the number of position/velocity vectors to */
/* process from the files used as input to this program. Each file */
/* must, at a minimum, contain this number of vectors.            */
/*******************************************************************/


#define NUMBER_OF_VECTORS 139


/*************************************        **********************/
/* X_P_49247 and Y_P_49247 are the angular c    .acements in arc seconds */
/* of the Celestial Ephemeris Pole (CEP) from the Conventional    */
/* Terrestial Pole (CTP) (CTP=ECEF_WGS_84 Z axis) in effect at Modified */
```

```
/* Julian Date (MJD) 49247.0. Likewise, X_P_49248 and Y_P_49248 are      */
/* these values at MJD 49248.0. UT1_UTC_49247 and UT1_UTC_49248 are the  */
/* differences in seconds between the UT1 and UTC timescales at MJD      */
/* 49247.0 and MJD 49248.0 respectively. Ex: UT1-UT1_UTC_49247=UTC.      */
/* TAI_UTC_OFFSET is the integer difference in seconds between           */
/* International Atomic Time (TAI) and UTC. Ex: TAI-TAI_UTC_OFFSET=UTC.   */
/* This difference is exact, and changes at not less than six month      */
/* intervals through introduction of leap seconds. TDT_TAI_OFFSET is     */
/* the set difference in seconds between Terrestrial Dynamical Time       */
/* (TDT) and TAI. Ex: TDT-TDT_TAI_OFFSET=TAI. Future predicted, and      */
/* past observed values of x_p, y_p, & ut1_utc, as well as               */
/* TAI_UTC_OFFSET and TDT_TAI_OFFSET are published by the International   */
/* Earth Rotation Service (IERS) (in the U.S., by the National Earth     */
/* Orientation Service (NEOS)). The following were obtained by           */
/* directing an anonymous FTP into MAIA.USNO.NAVY.MIL to access the      */
/* SER7 directory.                                                        */
/***********************************************************************/

#define X_P_49247 -.097
#define Y_P_49247 .345
#define UT1_UTC_49247 .4570
#define X_P_49248 -.097
#define Y_P_49248 .347
#define UT1_UTC_49248 .4543
#define TAI_UTC_OFFSET 28.0
#define TDT_TAI_OFFSET 32.184


/***********************************************************************/
/* GPS_UTC_OFFSET is the inte    difference in seconds between GPS time */
/* and Coordinated Universal       (UTC). Ex: GPS-GPS_UTC_OFFSET=UTC.   */
/* For Standard Positioning Service (SPS), GPS time is accurate to      */
/* within 337 nanoseconds of UTC time after GPS_UTC_OFFSET is applied.  */
/* GPS_WEEK_NUMBER is the number of weeks elapsed since 06 JAN 80.      */
/* GPS_UTC_OFFSET and GPS_WEEK_NUMBER are transmitted in NAV-messages   */
/* from GPS Satellite Vehicles (SVs). JD_GPS_WEEK_0 is the Julian Date  */
/* at 0h 06 JAN 80 in UTC.                                              */
/***********************************************************************/

#define GPS_UTC_OFFSET 9.0
#define GPS_WEEK_NUMBER 714.0
#define JD_GPS_WEEK_0 2444244.5


/***********************************************************************/
/* JD_00_JAN_93 is the Julian Date at 0h 00 JAN 93 in UTC. JD_J2000 is  */
/* the Julian Date at Standard Epoch J2000.0 in Barycentric Dynamical   */
/* Time (TDB).                                                          */
```

```c
/****************************************************************/

#define JD_00_JAN_93 2448987.5
#define JD_J2000 2451545.0

#include <bcd.h>
#include <math.h>
#include <stdio.h>
#include <stdlib.h>

void nerror(char*);

FILE *j,*k;
int n,o,i;
long double
a_i[106][5],s_i[106][2],c_i[106][2],a_n[4][5],s_n[4][2],c_n[4][2];
long double m_0[3][3],t_0[NUMBER_OF_VECTORS][6];
long double v_0[NUMBER_OF_VECTORS][6],v_1[NUMBER_OF_VECTORS][6];

/*                                                            */
/* nerror is Numerical Recipes standard error handler         */
/*                                                            */

void nerror(char error_text[])
{
   fprintf(stderr,"Numerical Recipes run-time error...\n");
   fprintf(stderr,"%s\n",error_text);
   fprintf(stderr,"...now exiting to system...\n");
   _exit(1);
}


/****************************************************************/
/*                                                            */
/* a_i[][],s_i[][], and c_i[][] are the IAU 1980 Nutation series */
/* multipliers and coefficients (Table 3.222.1, Explanatory Supplement */
/* to the Astronomical Almanac, 1992).                        */
/* a_n[][],s_n[][], and c_n[][] are multipliers and coefficients for */
/* corrections to the IAU 1980 Nutation series (Table 3.224.1, */
/* Explanatory Supplement to the Astronomical Alamanac, 1992). */
/* m_0[][] is the transformation matrix from B1950.0 to J2000.0 a.k.a. */
/* DE118/LE62 to DE200/LE200 (E.M. Standish, Astronomy and Astrophysics */
/* 114, pp. 297-302, 1982).                                   */
/* t_0[i][0] is the time element of the ith state vector in seconds */
/* since the beginning of the week (GPS time). t_0[i][0]==t_0[i][1]. */
/* t_0[i][2] is the time element of the ith state vector in hours */
/* since the beginning of the year (UTC) (Consistent with the Orbiter */
```

```c
/* reference trajectory state vectors supplied by NASA JSC).          */
/* t_0[i][3] is the Julian Date of the ith state vector in TDB.       */
/* t_0[i][4] is the Julian Date at 0h on the day of interest in UT1.  */
/* t_0[i][5] is the time element of the ith state vector in seconds   */
/* since the beginning of the day (UTC).                              */
/* v_0[i][0], v_0[i][1], and v_0[i][2] are the X, Y, and Z cartesian  */
/* position coordinates, respectively, of the ith state vector in the */
/* ECEF_WGS_84 reference frame.                                       */
/* v_0[i][3], v_0[i][4], and v_0[i][5] are the X_DOT, Y_DOT, and Z_DOT */
/* cartesian velocity coordinates, respectively, of the ith state     */
/* vector in the ECEF_WGS_84 reference frame.                         */
/* v_1[i][0], v_1[i][1], and v_1[i][2] are the X, Y, and Z cartesian  */
/* position coordinates, respectively, of the ith state vector in the */
/* ECI_M50 reference frame.                                           */
/* v_1[i][2], v_1[i][4], and v_1[i][5] are the X_DOT, Y_DOT, and Z_DOT */
/* cartesian velocity coordinates, respectively, of the ith state     */
/* vector in the ECI_M50 reference frame.                             */
/*                                                                    */
/* ECEF_WGS_84 coordinates/GPS time are read from ASCII position and  */
/* velocity data files, created by the TANSPOST program, which extracts */
/* desired data from the binary data files created by the TANSIO      */
/* program on STS-51. Transformed ECI_M50 coordinates/UTC are written */
/* to an ASCII data file.                                             */
/* FILE SPECIFICATIONS: -ECEF_WGS_84 Position File Name = statep.001  */
/*                      -ECEF_WGS_84 Velocity File Name = statev.001  */
/*                      -ECI_M50 State Vector File Name = stateout.001 */
/*                      -Data appearing on the same line in both input */
/*                       files must correspond to the same time       */
/*                       element.                                     */
/*                      -The number of vectors in the input files must */
/*                       be equal to or greater than the number       */
/*                        appearing in the #define NUMBER_OF_VECTORS  */
/*                       preprocessor statement.                      */
/*                                                                    */
/* From TANS packet 41:                                               */
/* GPS_UTC_OFFSET = 9 seconds throughout the mission.                 */
/* GPS_WEEK_NUMBER = 714 starting 12 SEP 94                           */
/* GPS_WEEK_NUMBER = 715 starting 19 SEP 94                           */
/* The correct numbers must appear in the #define preprocessor        */
/* statements.                                                        */
/*                                                                    */
/* Coordinate conversion methodology:                                 */
/* ECI_M50 position = [abcpm_0]transpose * ECEF_WGS_84 position       */
/* ECI_M50 velocity = [ab_dotcpm_0]transpose * ECEF_WGS_84 position + */
/*       [abcpm_0]transpose * ECEF_WGS_84 velocity                    */
/* a, b, b_dot, c, p, and m_0 are rotation matrices for polar motion, */
```

```c
/* Earth rotation, rate of change of Earth rotation matrix, astronomic  */
/* nutation, general precession, and standard epoch conversion,         */
/* respectively. a through p are time varying 3x3 matrices, though only */
/* the rate of change of b is significant. m_0 is a constant 3x3        */
/* matrix.                                                              */
/*                                                                      */
/***********************************************************************/

void main(void)
{
    long double longest=0.0;
// Initialize Nutation series a_i[][] s_i[][] c_i[][] a_n[][] s_n[][] c_n[][]*
    for(n=0;n<106;n++)
    {
        for(o=0;o<5;o++)
        {
            a_i[n][o]=0.0;
        }
        for(o=0;o<2;o++)
        {
            s_i[n][o]=0.0;
            c_i[n][o]=0.0;
        }
    }
    for(n=0;n<4;n++)
    {
        for(o=0;o<5;o++)
        {
            a_n[n][o]=0.0;
        }
        for(o=0;o<2;o++)
        {
            s_n[n][o]=0.0;
            c_n[n][o]=0.0;
        }
    }
    a_i[0][4]=1.0; a_i[1][4]=2.0; a_i[2][0]=-2.0; a_i[2][2]=2.0; a_i[2][4]=1.0;
    a_i[3][0]=2.0; a_i[3][2]=-2.0; a_i[4][0]=-2.0; a_i[4][2]=2.0;
    a_i[4][4]=2.0; a_i[5][0]=1.0; a_i[5][1]=-1.0; a_i[5][3]=-1.0;
    a_i[6][1]=-2.0; a_i[6][2]=2.0; a_i[6][3]=-2.0; a_i[6][4]=1.0;
    a_i[7][0]=2.0; a_i[7][2]=-2.0; a_i[7][4]=1.0; a_i[8][2]=2.0;
    a_i[8][3]=-2.0; a_i[8][4]=2.0; a_i[9][1]=1.0; a_i[10][1]=1.0;
    a_i[10][2]=2.0; a_i[10][3]=-2.0; a_i[10][4]=2.0; a_i[11][1]=-1.0;
    a_i[11][2]=2.0; a_i[11][3]=-2.0; a_i[11][4]=2.0; a_i[12][2]=2.0;
    a_i[12][3]=-2.0; a_i[12][4]=1.0; a_i[13][0]=2.0; a_i[13][3]=-2.0;
    a_i[14][2]=2.0; a_i[14][3]=-2.0; a_i[15][1]=2.0; a_i[16][1]=1.0;
```

```
a_i[16][4]=1.0; a_i[17][1]=2.0; a_i[17][2]=2.0; a_i[17][3]=-2.0;
a_i[17][4]=2.0; a_i[18][1]=-1.0; a_i[18][4]=1.0; a_i[19][0]=-2.0;
a_i[19][3]=2.0; a_i[19][4]=1.0; a_i[20][1]=-1.0; a_i[20][2]=2.0;
a_i[20][3]=-2.0; a_i[20][4]=1.0; a_i[21][0]=2.0; a_i[21][3]=-2.0;
a_i[21][4]=1.0; a_i[22][1]=1.0; a_i[22][2]=2.0; a_i[22][3]=-2.0;
a_i[22][4]=1.0; a_i[23][0]=1.0; a_i[23][3]=-1.0; a_i[24][0]=2.0;
a_i[24][1]=1.0; a_i[24][3]=-2.0; a_i[25][2]=-2.0; a_i[25][3]=2.0;
a_i[25][4]=1.0; a_i[26][1]=1.0; a_i[26][2]=-2.0; a_i[26][3]=2.0;
a_i[27][1]=1.0; a_i[27][4]=2.0; a_i[28][0]=-1.0; a_i[28][3]=1.0;
a_i[28][4]=1.0; a_i[29][1]=1.0; a_i[29][2]=2.0; a_i[29][3]=-2.0;
a_i[30][2]=2.0; a_i[30][4]=2.0; a_i[31][0]=1.0; a_i[32][2]=2.0;
a_i[32][4]=1.0; a_i[33][0]=1.0; a_i[33][2]=2.0; a_i[33][4]=2.0;
a_i[34][0]=1.0; a_i[34][3]=-2.0; a_i[35][0]=-1.0; a_i[35][2]=2.0;
a_i[35][4]=2.0; a_i[36][3]=2.0; a_i[37][0]=1.0; a_i[37][4]=1.0;
a_i[38][0]=-1.0; a_i[38][4]=1.0; a_i[39][0]=-1.0; a_i[39][2]=2.0;
a_i[39][3]=2.0; a_i[39][4]=2.0; a_i[40][0]=1.0; a_i[40][2]=2.0;
a_i[40][4]=1.0; a_i[41][2]=2.0; a_i[41][3]=2.0; a_i[41][4]=2.0;
a_i[42][0]=2.0; a_i[43][0]=1.0; a_i[43][2]=2.0; a_i[43][3]=-2.0;
a_i[43][4]=2.0; a_i[44][0]=2.0; a_i[44][2]=2.0; a_i[44][4]=2.0;
a_i[45][2]=2.0; a_i[46][0]=-1.0; a_i[46][2]=2.0; a_i[46][4]=1.0;
a_i[47][0]=-1.0; a_i[47][3]=2.0; a_i[47][4]=1.0; a_i[48][0]=1.0;
a_i[48][3]=-2.0; a_i[48][4]=1.0; a_i[49][0]=-1.0; a_i[49][2]=2.0;
a_i[49][3]=2.0; a_i[49][4]=1.0; a_i[50][0]=1.0; a_i[50][1]=1.0;
a_i[50][3]=-2.0; a_i[51][1]=1.0; a_i[51][2]=2.0; a_i[51][4]=2.0;
a_i[52][1]=-1.0; a_i[52][2]=2.0; a_i[52][4]=2.0; a_i[53][0]=1.0;
a_i[53][2]=2.0; a_i[53][3]=2.0; a_i[53][4]=2.0; a_i[54][0]=1.0;
a_i[54][3]=2.0; a_i[55][0]=2.0; a_i[55][2]=2.0; a_i[55][3]=-2.0;
a_i[55][4]=2.0; a_i[56][3]=2.0; a_i[56][4]=1.0; a_i[57][2]=2.0;
a_i[57][3]=2.0; a_i[57][4]=1.0; a_i[58][0]=1.0; a_i[58][2]=2.0;
a_i[58][3]=-2.0; a_i[58][4]=1.0; a_i[59][3]=-2.0; a_i[59][4]=1.0;
a_i[60][0]=1.0; a_i[60][1]=-1.0; a_i[61][0]=2.0; a_i[61][2]=2.0;
a_i[61][4]=1.0; a_i[62][1]=1.0; a_i[62][3]=-2.0; a_i[63][0]=1.0;
a_i[63][2]=-2.0; a_i[64][3]=1.0; a_i[65][0]=1.0; a_i[65][1]=1.0;
a_i[66][0]=1.0; a_i[66][2]=2.0; a_i[67][0]=1.0; a_i[67][1]=-1.0;
a_i[67][2]=2.0; a_i[67][4]=2.0; a_i[68][0]=-1.0; a_i[68][1]=-1.0;
a_i[68][2]=2.0; a_i[68][3]=2.0; a_i[68][4]=2.0; a_i[69][0]=-2.0;
a_i[69][4]=1.0; a_i[70][0]=3.0; a_i[70][2]=2.0; a_i[70][4]=2.0;
a_i[71][1]=-1.0; a_i[71][2]=2.0; a_i[71][3]=2.0; a_i[71][4]=2.0;
a_i[72][0]=1.0; a_i[72][1]=1.0; a_i[72][2]=2.0; a_i[72][4]=2.0;
a_i[73][0]=-1.0; a_i[73][2]=2.0; a_i[73][3]=-2.0; a_i[73][4]=1.0;
a_i[74][0]=2.0; a_i[74][4]=1.0; a_i[75][0]=1.0; a_i[75][4]=2.0;
a_i[76][0]=3.0; a_i[77][2]=2.0; a_i[77][3]=1.0; a_i[77][4]=2.0;
a_i[78][0]=-1.0; a_i[78][4]=2.0; a_i[79][0]=1.0; a_i[79][3]=-4.0;
a_i[80][0]=-2.0; a_i[80][2]=2.0; a_i[80][3]=2.0; a_i[80][4]=2.0;
a_i[81][0]=-1.0; a_i[81][2]=2.0; a_i[81][3]=4.0; a_i[81][4]=2.0;
a_i[82][0]=2.0; a_i[82][3]=-4.0; a_i[83][0]=1.0; a_i[83][1]=1.0;
```

```
a_i[83][2]=2.0; a_i[83][3]=-2.0; a_i[83][4]=2.0; a_i[84][0]=1.0;
a_i[84][2]=2.0; a_i[84][3]=2.0; a_i[84][4]=1.0; a_i[85][0]=-2.0;
a_i[85][2]=2.0; a_i[85][3]=4.0; a_i[85][4]=2.0; a_i[86][0]=-1.0;
a_i[86][2]=4.0; a_i[86][4]=2.0; a_i[87][0]=1.0; a_i[87][1]=-1.0;
a_i[87][3]=-2.0; a_i[88][0]=2.0; a_i[88][2]=2.0; a_i[88][3]=-2.0;
a_i[88][4]=1.0; a_i[89][0]=2.0; a_i[89][2]=2.0; a_i[89][3]=2.0;
a_i[89][4]=2.0; a_i[90][0]=1.0; a_i[90][3]=2.0; a_i[90][4]=1.0;
a_i[91][2]=4.0; a_i[91][3]=-2.0; a_i[91][4]=2.0; a_i[92][0]=3.0;
a_i[92][2]=2.0; a_i[92][3]=-2.0; a_i[92][4]=2.0; a_i[93][0]=1.0;
a_i[93][2]=2.0; a_i[93][3]=-2.0; a_i[94][1]=1.0; a_i[94][2]=2.0;
a_i[94][4]=1.0; a_i[95][0]=-1.0; a_i[95][1]=-1.0; a_i[95][3]=2.0;
a_i[95][4]=1.0; a_i[96][2]=-2.0; a_i[96][4]=1.0; a_i[97][2]=2.0;
a_i[97][3]=-1.0; a_i[97][4]=2.0; a_i[98][1]=1.0; a_i[98][3]=2.0;
a_i[99][0]=1.0; a_i[99][2]=-2.0; a_i[99][3]=-2.0; a_i[100][1]=-1.0;
a_i[100][2]=2.0; a_i[100][4]=1.0; a_i[101][0]=1.0; a_i[101][1]=1.0;
a_i[101][3]=-2.0; a_i[101][4]=1.0; a_i[102][0]=1.0; a_i[102][2]=-2.0;
a_i[102][3]=2.0; a_i[103][0]=2.0; a_i[103][3]=2.0; a_i[104][2]=2.0;
a_i[104][3]=4.0; a_i[104][4]=2.0; a_i[105][1]=1.0; a_i[105][3]=1.0;
s_i[0][0]=-171996.0; s_i[0][1]=-174.2; s_i[1][0]=2062.0; s_i[1][1]=.2;
s_i[2][0]=46.0; s_i[3][0]=11.0; s_i[4][0]=-3.0; s_i[5][0]=-3.0;
s_i[6][0]=-2.0; s_i[7][0]=1.0; s_i[8][0]=-13187.0; s_i[8][1]=-1.6;
s_i[9][0]=1426.0; s_i[9][1]=-3.4; s_i[10][0]=-517.0; s_i[10][1]=1.2;
s_i[11][0]=217; s_i[11][1]=-.5; s_i[12][0]=129.0; s_i[12][1]=.1;
s_i[13][0]=48.0; s_i[14][0]=-22.0; s_i[15][0]=17.0; s_i[15][1]=-.1;
s_i[16][0]=-15.0; s_i[17][0]=-16.0; s_i[17][1]=.1; s_i[18][0]=-12.0;
s_i[19][0]=-6.0; s_i[20][0]=-5.0; s_i[21][0]=4.0; s_i[22][0]=4.0;
s_i[23][0]=-4.0; s_i[24][0]=1.0; s_i[25][0]=1.0; s_i[26][0]=-1.0;
s_i[27][0]=1.0; s_i[28][0]=1.0; s_i[29][0]=-1.0; s_i[30][0]=-2274.0;
s_i[30][1]=-.2; s_i[31][0]=712.0; s_i[31][1]=.1; s_i[32][0]=-386.0;
s_i[32][1]=-.4; s_i[33][0]=-301.0; s_i[34][0]=-158.0; s_i[35][0]=123.0;
s_i[36][0]=63.0; s_i[37][0]=63.0; s_i[37][1]=.1; s_i[38][0]=-58.0;
s_i[38][1]=-.1; s_i[39][0]=-59.0; s_i[40][0]=-51.0; s_i[41][0]=-38.0;
s_i[42][0]=29.0; s_i[43][0]=29.0; s_i[44][0]=-31.0; s_i[45][0]=26.0;
s_i[46][0]=21.0; s_i[47][0]=16.0; s_i[48][0]=-13.0; s_i[49][0]=-10.0;
s_i[50][0]=-7.0; s_i[51][0]=7.0; s_i[52][0]=-7.0; s_i[53][0]=-8.0;
s_i[54][0]=6.0; s_i[55][0]=6.0; s_i[56][0]=-6.0; s_i[57][0]=-7.0;
s_i[58][0]=6.0; s_i[59][0]=-5.0; s_i[60][0]=5.0; s_i[61][0]=-5.0;
s_i[62][0]=-4.0; s_i[63][0]=4.0; s_i[64][0]=-4.0; s_i[65][0]=-3.0;
s_i[66][0]=3.0; s_i[67][0]=-3.0; s_i[68][0]=-3.0; s_i[69][0]=-2.0;
s_i[70][0]=-3.0; s_i[71][0]=-3.0; s_i[72][0]=2.0; s_i[73][0]=-2.0;
s_i[74][0]=2.0; s_i[75][0]=-2.0; s_i[76][0]=2.0; s_i[77][0]=2.0;
s_i[78][0]=1.0; s_i[79][0]=-1.0; s_i[80][0]=1.0; s_i[81][0]=-2.0;
s_i[82][0]=-1.0; s_i[83][0]=1.0; s_i[84][0]=-1.0; s_i[85][0]=-1.0;
s_i[86][0]=1.0; s_i[87][0]=1.0; s_i[88][0]=1.0; s_i[89][0]=-1.0;
s_i[90][0]=-1.0; s_i[91][0]=1.0; s_i[92][0]=1.0; s_i[93][0]=-1.0;
s_i[94][0]=1.0; s_i[95][0]=1.0; s_i[96][0]=-1.0; s_i[97][0]=-1.0;
```

```c
s_i[98][0]=-1.0; s_i[99][0]=-1.0; s_i[100][0]=-1.0; s_i[101][0]=-1.0;
s_i[102][0]=-1.0; s_i[103][0]=1.0; s_i[104][0]=-1.0; s_i[105][0]=1.0;
c_i[0][0]=92025.0; c_i[0][1]=8.9; c_i[1][0]=-895.0; c_i[1][1]=.5;
c_i[2][0]=-24.0; c_i[4][0]=1.0; c_i[6][0]=1.0; c_i[8][0]=5736.0;
c_i[8][1]=-3.1; c_i[9][0]=54.0; c_i[9][1]=-.1; c_i[10][0]=224.0;
c_i[10][1]=-.6; c_i[11][0]=-95.0; c_i[11][1]=.3; c_i[12][0]=-70.0;
c_i[13][0]=1.0; c_i[16][0]=9.0; c_i[17][0]=7.0; c_i[18][0]=6.0;
c_i[19][0]=3.0; c_i[20][0]=3.0; c_i[21][0]=-2.0; c_i[22][0]=-2.0;
c_i[30][0]=977.0; c_i[30][1]=-.5; c_i[31][0]=-7.0; c_i[32][0]=200.0;
c_i[33][0]=129.0; c_i[33][1]=-.1; c_i[34][0]=-1.0; c_i[35][0]=-53.0;
c_i[36][0]=-2.0; c_i[37][0]=-33.0; c_i[38][0]=32.0; c_i[39][0]=26.0;
c_i[40][0]=27.0; c_i[41][0]=16.0; c_i[42][0]=-1.0; c_i[43][0]=-12.0;
c_i[44][0]=13.0; c_i[45][0]=-1.0; c_i[46][0]=-10.0; c_i[47][0]=-8.0;
c_i[48][0]=7.0; c_i[49][0]=5.0; c_i[51][0]=-3.0; c_i[52][0]=3.0;
c_i[53][0]=3.0; c_i[55][0]=-3.0; c_i[56][0]=3.0; c_i[57][0]=3.0;
c_i[58][0]=-3.0; c_i[59][0]=3.0; c_i[61][0]=3.0; c_i[67][0]=1.0;
c_i[68][0]=1.0; c_i[69][0]=1.0; c_i[70][0]=1.0; c_i[71][0]=1.0;
c_i[72][0]=-1.0; c_i[73][0]=1.0; c_i[74][0]=-1.0; c_i[75][0]=1.0;
c_i[77][0]=-1.0; c_i[78][0]=-1.0; c_i[80][0]=-1.0; c_i[81][0]=1.0;
c_i[83][0]=-1.0; c_i[84][0]=1.0; c_i[85][0]=1.0; c_i[88][0]=-1.0;
a_n[0][4]=1.0; a_n[1][1]=1.0; a_n[2][2]=2.0; a_n[2][3]=-2.0; a_n[2][4]=2.0;
a_n[3][2]=2.0; a_n[3][4]=2.0;
s_n[0][0]=-725.0; s_n[0][1]=224.0; s_n[1][0]=523.0; s_n[1][1]=-24.0;
s_n[2][0]=102.0; s_n[2][1]=-47.0; s_n[3][0]=-81.0;
c_n[0][0]=417.0; c_n[0][1]=213.0; c_n[1][0]=61.0; c_n[1][1]=208.0;
c_n[2][0]=-118.0; c_n[2][1]=-41.0; c_n[3][1]=32.0;
// Initialize the DE118/LE62 to DE200/LE200 transformation matrix*************
m_0[0][0]=.9999256791774783; m_0[0][1]=-.0111815116768724;
m_0[0][2]=-.0048590038154553; m_0[1][0]=.0111815116959975;
m_0[1][1]=.9999374845751042; m_0[1][2]=-.0000271625775175;
m_0[2][0]=.004859003771445; m_0[2][1]=-.000027170449221;
m_0[2][2]=.9999881946023742;
// Read ECEF_WGS_84 state vectors from external files ********************
if((j=fopen("statep.001","r"))==NULL)
{
    fclose(j);
    nerror("unable to open statep.001 in main()");
}
if((k=fopen("statev.001","r"))==NULL)
{
    fclose(k);
    nerror("unable to open statev.001 in main()");
}
for(i=0;i<NUMBER_OF_VECTORS;i++)
{
    fscanf(j,"%Lf %Lf %Lf %Lf\n",&t_0[i][0],&v_0[i][0],&v_0[i][1],
```

```c
                &v_0[i][2]);
        fscanf(k,"%Lf %Lf %Lf %Lf\n",&t_0[i][1],&v_0[i][3],&v_0[i][4],
                &v_0[i][5]);
        t_0[i][0]*=1000000.0; v_0[i][0]*=1000000.0; v_0[i][1]*=1000000.0;
v_0[i][2]*=1000000.0;
        t_0[i][1]*=1000.0; v_0[i][3]*=1000.0; v_0[i][4]*=1000.0;
v_0[i][5]*=1000.0;
        t_0[i][0]+=522009.0; t_0[i][1]+=522009.0;
        if(i<5)
        {
            printf("%Lf %Lf\n",t_0[i][0],t_0[i][1]);
        }
        if(t_0[i][0]!=t_0[i][1])
        {
            nerror("time element mismatch in main()");
        }
    }
    if(fclose(j)==EOF)
    {
        nerror("unable to close statep.001 in main()");
    }
    if(fclose(k)==EOF)
    {
        nerror("unable to close statev.001 in main()");
    }
// Main loop to calculate time, rotation matrices, and transformations********
    for(i=0;i<NUMBER_OF_VECTORS;i++)
    {
        bcd tdt;
        long double g,integer,fraction;
        long double t,zeta,z,theta,p[3][3];
        long double l,l_prime,f,d,omega,epsilon_bar,epsilon,c[3][3];
        long double ut1_utc_interpolated,delta_psi=0.0,delta_epsilon=0.0;
        long double t_u,h_0,delta_h,omega_star,lambda,b[3][3],b_dot[3][3];
        long double x_p_interpolated,y_p_interpolated,a[3][3];
        long double pm_0[3][3],cpm_0[3][3],bcpm_0[3][3],b_dotcpm_0[3][3];
        long double abcpm_0[3][3],ab_dotcpm_0[3][3];
        long double abcpm_0_transpose[3][3],ab_dotcpm_0_transpose[3][3];
// Convert time (t_0[i][0] & t_0[i][1]) to UTC/TDB/UT1 (t_0[i][2] - t_0[i][5])
    // Compute UTC time in hours since the beginning of the year***************
        t_0[i][2]=((t_0[i][0]/DAY)+(GPS_WEEK_NUMBER*7.0)+JD_GPS_WEEK_0-
                (GPS_UTC_OFFSET/DAY)-JD_00_JAN_93)*24.0;
    // Compute Julian Date in TDT time rounded to two decimal places***********
        tdt=bcd(((t_0[i][0]/DAY)+(GPS_WEEK_NUMBER*7.0)+JD_GPS_WEEK_0-
            (GPS_UTC_OFFSET/DAY)+(TAI_UTC_OFFSET/DAY)+(TDT_TAI_OFFSET/DAY)),2);
    // Compute mean anomaly of the Earth in its orbit************************
```

```c
      g=357.53+(.9856003*(real(tdt)-JD_J2000));
   // Compute Julian Date in TDB time*******************************************
      t_0[i][3]=(t_0[i][0]/DAY)+(GPS_WEEK_NUMBER*7.0)+JD_GPS_WEEK_0-
               (GPS_UTC_OFFSET/DAY)+(TAI_UTC_OFFSET/DAY)+(TDT_TAI_OFFSET/
               DAY)+(((.001658*sinl(g*DEG_TO_RAD))+(.000014*sinl(2.0*g*
               DEG_TO_RAD)))/DAY);
   // Compute Julian Date in UT1 time*******************************************
      if(((t_0[i][0]/DAY)+(GPS_WEEK_NUMBER*7.0)+JD_GPS_WEEK_0-
(GPS_UTC_OFFSET/
         DAY)-2400000.5)<49248.0)
      {
         t_0[i][4]=(t_0[i][0]/DAY)+(GPS_WEEK_NUMBER*7.0)+JD_GPS_WEEK_0-
                  (GPS_UTC_OFFSET/DAY)+(UT1_UTC_49247/DAY);
      }
      if(((t_0[i][0]/DAY)+(GPS_WEEK_NUMBER*7.0)+JD_GPS_WEEK_0-
(GPS_UTC_OFFSET/
         DAY)-2400000.5)>=49248.0)
      {
         t_0[i][4]=(t_0[i][0]/DAY)+(GPS_WEEK_NUMBER*7.0)+JD_GPS_WEEK_0-
                  (GPS_UTC_OFFSET/DAY)+(UT1_UTC_49248/DAY);
      }
   // Compute Julian Date in UT1 time at the beginning of the day*************
      fraction=modfl(t_0[i][4],&integer);
      if(fraction<.5)
      {
         t_0[i][4]=integer-.5;
      }
      if(fraction>.5)
      {
         t_0[i][4]=integer+.5;
      }
   // Compute UTC time in seconds since the beginning of the day**************
      fraction=modfl(((t_0[i][0]-GPS_UTC_OFFSET)/DAY),&integer);
      t_0[i][5]=(t_0[i][0]-GPS_UTC_OFFSET)-(integer*DAY);
// Calculate General Precession Rotation Matrix p[][]*************************
   // Calculate number of centuries of TDB elapsed since J2000.0**************
      t=(t_0[i][3]-JD_J2000)/CENTURY;
   // Calculate Accumulated Precession Angles Adopted by IAU 1976*************
      zeta=((2306.2181*t)+(.30188*t*t)+(.017998*t*t*t))*ARCSEC_TO_RAD;
      z=((2306.2181*t)+(1.09468*t*t)+(.018203*t*t*t))*ARCSEC_TO_RAD;
      theta=((2004.3109*t)-(.42665*t*t)-(.041833*t*t*t))*ARCSEC_TO_RAD;
      p[0][0]=(cosl(z)*cosl(theta)*cosl(zeta))-(sinl(z)*sinl(zeta));
      p[0][1]=(-cosl(z)*cosl(theta)*sinl(zeta))-(sinl(z)*cosl(zeta));
      p[0][2]=-cosl(z)*sinl(theta);
      p[1][0]=(sinl(z)*cosl(theta)*cosl(zeta))+(cosl(z)*sinl(zeta));
      p[1][1]=(-sinl(z)*cosl(theta)*sinl(zeta))+(cosl(z)*cosl(zeta));
```

```
        p[1][2]=-sinl(z)*sinl(theta); p[2][0]=sinl(theta)*cosl(zeta);
        p[2][1]=-sinl(theta)*sinl(zeta); p[2][2]=cosl(theta);
// Calculate Astronomical Nutation Rotation Matrix c[][]*********************
    // Calculate IAU 1980 Nutation set of Fundamental Arguments****************
        l=(485866.733+(((1325.0*ONE_REV)+715922.633)*t)+(31.31*t*t)+(.064*t*t*
            t))*ARCSEC_TO_RAD;
        l_prime=(1287099.804+(((99.0*ONE_REV)+1292581.244)*t)-(.577*t*t)-(.012*
                t*t*t))*ARCSEC_TO_RAD;
        f=(335778.877+(((1342.0*ONE_REV)+295263.137)*t)-(13.257*t*t)+(.011*t*t*
            t))*ARCSEC_TO_RAD;
        d=(1072261.307+(((1236.0*ONE_REV)+1105601.328)*t)-
(6.891*t*t)+(.019*t*t*
            t))*ARCSEC_TO_RAD;
        omega=(450160.28-(((5.0*ONE_REV)+482890.539)*t)+(7.455*t*t)+(.008*t*t*
            t))*ARCSEC_TO_RAD;
    // Calculate Mean Obliquity of the Ecliptic*******************************
        epsilon_bar=(84381.448-(46.815*t)-(.00059*t*t)+(.001813*t*t*t))*
            ARCSEC_TO_RAD;
    // Calculate Angle of Nutation in Longitude******************************
        for(n=0;n<106;n++)
        {
            long double delta_psi_i;
            delta_psi_i=(s_i[n][0]+(s_i[n][1]*t))*sinl((a_i[n][0]*l)+(a_i[n][1]*

l_prime)+(a_i[n][2]*f)+(a_i[n][3]*d)+(a_i[n][4]*omega));
            delta_psi+=(delta_psi_i*.0001*ARCSEC_TO_RAD);
        }
    // Calculate and apply Correction to Angle of Nutation in Longitude********
        for(n=0;n<4;n++)
        {
            long double delta_psi_c;
            delta_psi_c=(s_n[n][0]*sinl((a_n[n][0]*l)+(a_n[n][1]*l_prime)+

(a_n[n][2]*f)+(a_n[n][3]*d)+(a_n[n][4]*omega)))+

(c_n[n][0]*cosl((a_n[n][0]*l)+(a_n[n][1]*l_prime)+

(a_n[n][2]*f)+(a_n[n][3]*d)+(a_n[n][4]*omega)));
            delta_psi+=(delta_psi_c*.00001*ARCSEC_TO_RAD);
        }
    // Calculate Angle of Nutation in Obliquity******************************
        for(n=0;n<106;n++)
        {
            long double delta_epsilon_i;
            delta_epsilon_i=(c_i[n][0]+(c_i[n][1]*t))*cosl((a_i[n][0]*l)+
```

```
(a_i[n][1]*l_prime)+(a_i[n][2]*f)+(a_i[n][3]*d)+

(a_i[n][4]*omega));
        delta_epsilon+=(delta_epsilon_i*.0001*ARCSEC_TO_RAD);
    }
    // Calculate and apply Correction to Angle of Nutation in Obliquity********
    for(n=0;n<4;n++)
    {
        long double delta_epsilon_c;
        delta_epsilon_c=(c_n[n][1]*cosl((a_n[n][0]*l)+(a_n[n][1]*l_prime)+

(a_n[n][2]*f)+(a_n[n][3]*d)+(a_n[n][4]*omega)))+

(s_n[n][1]*sinl((a_n[n][0]*l)+(a_n[n][1]*l_prime)+

(a_n[n][2]*f)+(a_n[n][3]*d)+(a_n[n][4]*omega)));
        delta_epsilon+=(delta_epsilon_c*.00001*ARCSEC_TO_RAD);
    }
    // Calculate True Obliquity of the Ecliptic*******************************
    epsilon=epsilon_bar+delta_epsilon;
    c[0][0]=cosl(delta_psi);  c[0][1]=-sinl(delta_psi)*cosl(epsilon_bar);
    c[0][2]=-sinl(delta_psi)*sinl(epsilon_bar);
    c[1][0]=cosl(epsilon)*sinl(delta_psi);
    c[1][1]=(cosl(epsilon)*cosl(delta_psi)*cosl(epsilon_bar))+
            (sinl(epsilon)*sinl(epsilon_bar));
    c[1][2]=(cosl(epsilon)*cosl(delta_psi)*sinl(epsilon_bar))-
            (sinl(epsilon)*cosl(epsilon_bar));
    c[2][0]=sinl(epsilon)*sinl(delta_psi);
    c[2][1]=(sinl(epsilon)*cosl(delta_psi)*cosl(epsilon_bar))-
            (cosl(epsilon)*sinl(epsilon_bar));
    c[2][2]=(sinl(epsilon)*cosl(delta_psi)*sinl(epsilon_bar))+
            (cosl(epsilon)*cosl(epsilon_bar));
// Calculate Earth Rotation (Sidereal Time) Matrix b[][]********************
    // Interpolate values of ut1_utc****************************************
    ut1_utc_interpolated=(((((t_0[i][2]/24.0)+JD_00_JAN_93)-2400000.5)-

49247.0)*(UT1_UTC_49248-UT1_UTC_49247))+
                                                UT1_UTC_49247;
    // Calculate number of centuries of UT elapsed since 12h 01 JAN 2000 UT1***
    t_u=(t_0[i][4]-2451545.0)/CENTURY;
    // Calculate Greenwich Mean Sidereal Time at 0h UT1 of day of interest*****
    h_0=(24110.54841+(8640184.812866*t_u)+(.093104*t_u*t_u)-(.0000062*t_u*
        t_u*t_u))*SEC_TO_RAD;
    // Calculate the Equation of the Equnoxes********************************
    delta_h=atanl(cosl(epsilon)*tanl(delta_psi));
    // Calculate Earth rotation rate in a precessing reference frame***********
```

75

```
        omega_star=OMEGA_PRIME+.000000000007086+(.0000000000000043*t_u);
    // Calculate Longitude of the Zero Meridian from the true vernal equinox***
        lambda=h_0+delta_h+(omega_star*(t_0[i][5]+ut1_utc_interpolated));
        b[0][0]=cosl(lambda); b[0][1]=sinl(lambda); b[0][2]=0.0;
        b[1][0]=-sinl(lambda); b[1][1]=cosl(lambda); b[1][2]=0.0; b[2][0]=0.0;
        b[2][1]=0.0; b[2][2]=1.0;
// Calculate Change in Earth Rotation (Sidereal Time) Matrix b_dot[][]********
        b_dot[0][0]=-omega_star*sinl(lambda);
        b_dot[0][1]=omega_star*cosl(lambda); b_dot[0][2]=0.0;
        b_dot[1][0]=-omega_star*cosl(lambda);
        b_dot[1][1]=-omega_star*sinl(lambda); b_dot[1][2]=0.0; b_dot[2][0]=0.0;
        b_dot[2][1]=0.0; b_dot[2][2]=0.0;
// Calculate Polar Motion Rotation Matrix a[][]********************************
    // Interpolate values of x_p & y_p and convert to radians******************
        x_p_interpolated=((((((t_0[i][2]/24.0)+JD_00_JAN_93)-2400000.5)-

49247.0)*(X_P_49248-X_P_49247))+X_P_49247)*
                                                            ARCSEC_TO_RAD;
        y_p_interpolated=((((((t_0[i][2]/24.0)+JD_00_JAN_93)-2400000.5)-

49247.0)*(Y_P_49248-Y_P_49247))+Y_P_49247)*
                                                            ARCSEC_TO_RAD;
        a[0][0]=cosl(x_p_interpolated);
        a[0][1]=sinl(x_p_interpolated)*sinl(y_p_interpolated);
        a[0][2]=sinl(x_p_interpolated)*cosl(y_p_interpolated); a[1][0]=0.0;
        a[1][1]=cosl(y_p_interpolated); a[1][2]=-sinl(y_p_interpolated);
        a[2][0]=-sinl(x_p_interpolated);
        a[2][1]=cosl(x_p_interpolated)*sinl(y_p_interpolated);
        a[2][2]=cosl(x_p_interpolated)*cosl(y_p_interpolated);
// Perform matrix multiplication to form transformation matrices**************
        pm_0[0][0]=(p[0][0]*m_0[0][0])+(p[0][1]*m_0[1][0])+(p[0][2]*m_0[2][0]);
        pm_0[0][1]=(p[0][0]*m_0[0][1])+(p[0][1]*m_0[1][1])+(p[0][2]*m_0[2][1]);
        pm_0[0][2]=(p[0][0]*m_0[0][2])+(p[0][1]*m_0[1][2])+(p[0][2]*m_0[2][2]);
        pm_0[1][0]=(p[1][0]*m_0[0][0])+(p[1][1]*m_0[1][0])+(p[1][2]*m_0[2][0]);
        pm_0[1][1]=(p[1][0]*m_0[0][1])+(p[1][1]*m_0[1][1])+(p[1][2]*m_0[2][1]);
        pm_0[1][2]=(p[1][0]*m_0[0][2])+(p[1][1]*m_0[1][2])+(p[1][2]*m_0[2][2]);
        pm_0[2][0]=(p[2][0]*m_0[0][0])+(p[2][1]*m_0[1][0])+(p[2][2]*m_0[2][0]);
        pm_0[2][1]=(p[2][0]*m_0[0][1])+(p[2][1]*m_0[1][1])+(p[2][2]*m_0[2][1]);
        pm_0[2][2]=(p[2][0]*m_0[0][2])+(p[2][1]*m_0[1][2])+(p[2][2]*m_0[2][2]);
        cpm_0[0][0]=(c[0][0]*pm_0[0][0])+(c[0][1]*pm_0[1][0])+(c[0][2]*
                    pm_0[2][0]);
        cpm_0[0][1]=(c[0][0]*pm_0[0][1])+(c[0][1]*pm_0[1][1])+(c[0][2]*
                    pm_0[2][1]);
        cpm_0[0][2]=(c[0][0]*pm_0[0][2])+(c[0][1]*pm_0[1][2])+(c[0][2]*
                    pm_0[2][2]);
        cpm_0[1][0]=(c[1][0]*pm_0[0][0])+(c[1][1]*pm_0[1][0])+(c[1][2]*
```

```
                 pm_0[2][0]);
cpm_0[1][1]=(c[1][0]*pm_0[0][1])+(c[1][1]*pm_0[1][1])+(c[1][2]*
                 pm_0[2][1]);
cpm_0[1][2]=(c[1][0]*pm_0[0][2])+(c[1][1]*pm_0[1][2])+(c[1][2]*
                 pm_0[2][2]);
cpm_0[2][0]=(c[2][0]*pm_0[0][0])+(c[2][1]*pm_0[1][0])+(c[2][2]*
                 pm_0[2][0]);
cpm_0[2][1]=(c[2][0]*pm_0[0][1])+(c[2][1]*pm_0[1][1])+(c[2][2]*
                 pm_0[2][1]);
cpm_0[2][2]=(c[2][0]*pm_0[0][2])+(c[2][1]*pm_0[1][2])+(c[2][2]*
                 pm_0[2][2]);
bcpm_0[0][0]=(b[0][0]*cpm_0[0][0])+(b[0][1]*cpm_0[1][0])+(b[0][2]*
                 cpm_0[2][0]);
bcpm_0[0][1]=(b[0][0]*cpm_0[0][1])+(b[0][1]*cpm_0[1][1])+(b[0][2]*
                 cpm_0[2][1]);
bcpm_0[0][2]=(b[0][0]*cpm_0[0][2])+(b[0][1]*cpm_0[1][2])+(b[0][2]*
                 cpm_0[2][2]);
bcpm_0[1][0]=(b[1][0]*cpm_0[0][0])+(b[1][1]*cpm_0[1][0])+(b[1][2]*
                 cpm_0[2][0]);
bcpm_0[1][1]=(b[1][0]*cpm_0[0][1])+(b[1][1]*cpm_0[1][1])+(b[1][2]*
                 cpm_0[2][1]);
bcpm_0[1][2]=(b[1][0]*cpm_0[0][2])+(b[1][1]*cpm_0[1][2])+(b[1][2]*
                 cpm_0[2][2]);
bcpm_0[2][0]=(b[2][0]*cpm_0[0][0])+(b[2][1]*cpm_0[1][0])+(b[2][2]*
                 cpm_0[2][0]);
bcpm_0[2][1]=(b[2][0]*cpm_0[0][1])+(b[2][1]*cpm_0[1][1])+(b[2][2]*
                 cpm_0[2][1]);
bcpm_0[2][2]=(b[2][0]*cpm_0[0][2])+(b[2][1]*cpm_0[1][2])+(b[2][2]*
                 cpm_0[2][2]);
b_dotcpm_0[0][0]=(b_dot[0][0]*cpm_0[0][0])+(b_dot[0][1]*cpm_0[1][0])+
                 (b_dot[0][2]*cpm_0[2][0]);
b_dotcpm_0[0][1]=(b_dot[0][0]*cpm_0[0][1])+(b_dot[0][1]*cpm_0[1][1])+
                 (b_dot[0][2]*cpm_0[2][1]);
b_dotcpm_0[0][2]=(b_dot[0][0]*cpm_0[0][2])+(b_dot[0][1]*cpm_0[1][2])+
                 (b_dot[0][2]*cpm_0[2][2]);
b_dotcpm_0[1][0]=(b_dot[1][0]*cpm_0[0][0])+(b_dot[1][1]*cpm_0[1][0])+
                 (b_dot[1][2]*cpm_0[2][0]);
b_dotcpm_0[1][1]=(b_dot[1][0]*cpm_0[0][1])+(b_dot[1][1]*cpm_0[1][1])+
                 (b_dot[1][2]*cpm_0[2][1]);
b_dotcpm_0[1][2]=(b_dot[1][0]*cpm_0[0][2])+(b_dot[1][1]*cpm_0[1][2])+
                 (b_dot[1][2]*cpm_0[2][2]);
b_dotcpm_0[2][0]=(b_dot[2][0]*cpm_0[0][0])+(b_dot[2][1]*cpm_0[1][0])+
                 (b_dot[2][2]*cpm_0[2][0]);
b_dotcpm_0[2][1]=(b_dot[2][0]*cpm_0[0][1])+(b_dot[2][1]*cpm_0[1][1])+
                 (b_dot[2][2]*cpm_0[2][1]);
b_dotcpm_0[2][2]=(b_dot[2][0]*cpm_0[0][2])+(b_dot[2][1]*cpm_0[1][2])+
```

```
                        (b_dot[2][2]*cpm_0[2][2]);
        abcpm_0[0][0]=(a[0][0]*bcpm_0[0][0])+(a[0][1]*bcpm_0[1][0])+(a[0][2]*
                        bcpm_0[2][0]);
        abcpm_0[0][1]=(a[0][0]*bcpm_0[0][1])+(a[0][1]*bcpm_0[1][1])+(a[0][2]*
                        bcpm_0[2][1]);
        abcpm_0[0][2]=(a[0][0]*bcpm_0[0][2])+(a[0][1]*bcpm_0[1][2])+(a[0][2]*
                        bcpm_0[2][2]);
        abcpm_0[1][0]=(a[1][0]*bcpm_0[0][0])+(a[1][1]*bcpm_0[1][0])+(a[1][2]*
                        bcpm_0[2][0]);
        abcpm_0[1][1]=(a[1][0]*bcpm_0[0][1])+(a[1][1]*bcpm_0[1][1])+(a[1][2]*
                        bcpm_0[2][1]);
        abcpm_0[1][2]=(a[1][0]*bcpm_0[0][2])+(a[1][1]*bcpm_0[1][2])+(a[1][2]*
                        bcpm_0[2][2]);
        abcpm_0[2][0]=(a[2][0]*bcpm_0[0][0])+(a[2][1]*bcpm_0[1][0])+(a[2][2]*
                        bcpm_0[2][0]);
        abcpm_0[2][1]=(a[2][0]*bcpm_0[0][1])+(a[2][1]*bcpm_0[1][1])+(a[2][2]*
                        bcpm_0[2][1]);
        abcpm_0[2][2]=(a[2][0]*bcpm_0[0][2])+(a[2][1]*bcpm_0[1][2])+(a[2][2]*
                        bcpm_0[2][2]);
    ab_dotcpm_0[0][0]=(a[0][0]*b_dotcpm_0[0][0])+(a[0][1]*b_dotcpm_0[1][0])+
                        (a[0][2]*b_dotcpm_0[2][0]);
    ab_dotcpm_0[0][1]=(a[0][0]*b_dotcpm_0[0][1])+(a[0][1]*b_dotcpm_0[1][1])+
                        (a[0][2]*b_dotcpm_0[2][1]);
    ab_dotcpm_0[0][2]=(a[0][0]*b_dotcpm_0[0][2])+(a[0][1]*b_dotcpm_0[1][2])+
                        (a[0][2]*b_dotcpm_0[2][2]);
    ab_dotcpm_0[1][0]=(a[1][0]*b_dotcpm_0[0][0])+(a[1][1]*b_dotcpm_0[1][0])+
                        (a[1][2]*b_dotcpm_0[2][0]);
    ab_dotcpm_0[1][1]=(a[1][0]*b_dotcpm_0[0][1])+(a[1][1]*b_dotcpm_0[1][1])+
                        (a[1][2]*b_dotcpm_0[2][1]);
    ab_dotcpm_0[1][2]=(a[1][0]*b_dotcpm_0[0][2])+(a[1][1]*b_dotcpm_0[1][2])+
                        (a[1][2]*b_dotcpm_0[2][2]);
    ab_dotcpm_0[2][0]=(a[2][0]*b_dotcpm_0[0][0])+(a[2][1]*b_dotcpm_0[1][0])+
                        (a[2][2]*b_dotcpm_0[2][0]);
    ab_dotcpm_0[2][1]=(a[2][0]*b_dotcpm_0[0][1])+(a[2][1]*b_dotcpm_0[1][1])+
                        (a[2][2]*b_dotcpm_0[2][1]);
    ab_dotcpm_0[2][2]=(a[2][0]*b_dotcpm_0[0][2])+(a[2][1]*b_dotcpm_0[1][2])+
                        (a[2][2]*b_dotcpm_0[2][2]);
// Transpose the transformation matrices**************************************
        abcpm_0_transpose[0][0]=abcpm_0[0][0];
        abcpm_0_transpose[0][1]=abcpm_0[1][0];
        abcpm_0_transpose[0][2]=abcpm_0[2][0];
        abcpm_0_transpose[1][0]=abcpm_0[0][1];
        abcpm_0_transpose[1][1]=abcpm_0[1][1];
        abcpm_0_transpose[1][2]=abcpm_0[2][1];
        abcpm_0_transpose[2][0]=abcpm_0[0][2];
        abcpm_0_transpose[2][1]=abcpm_0[1][2];
```

```c
        abcpm_0_transpose[2][2]=abcpm_0[2][2];
        ab_dotcpm_0_transpose[0][0]=ab_dotcpm_0[0][0];
        ab_dotcpm_0_transpose[0][1]=ab_dotcpm_0[1][0];
        ab_dotcpm_0_transpose[0][2]=ab_dotcpm_0[2][0];
        ab_dotcpm_0_transpose[1][0]=ab_dotcpm_0[0][1];
        ab_dotcpm_0_transpose[1][1]=ab_dotcpm_0[1][1];
        ab_dotcpm_0_transpose[1][2]=ab_dotcpm_0[2][1];
        ab_dotcpm_0_transpose[2][0]=ab_dotcpm_0[0][2];
        ab_dotcpm_0_transpose[2][1]=ab_dotcpm_0[1][2];
        ab_dotcpm_0_transpose[2][2]=ab_dotcpm_0[2][2];
// Perform ECEF_WGS_84 to ECI_M50 position coordinate transformation*********
        v_1[i][0]=(abcpm_0_transpose[0][0]*v_0[i][0])+(abcpm_0_transpose[0][1]*
                v_0[i][1])+(abcpm_0_transpose[0][2]*v_0[i][2]);
        v_1[i][1]=(abcpm_0_transpose[1][0]*v_0[i][0])+(abcpm_0_transpose[1][1]*
                v_0[i][1])+(abcpm_0_transpose[1][2]*v_0[i][2]);
        v_1[i][2]=(abcpm_0_transpose[2][0]*v_0[i][0])+(abcpm_0_transpose[2][1]*
                v_0[i][1])+(abcpm_0_transpose[2][2]*v_0[i][2]);
// Perform ECEF_WGS_84 to ECI_M50 velocity coordinate transformation*********
        v_1[i][3]=(ab_dotcpm_0_transpose[0][0]*v_0[i][0])+
                (ab_dotcpm_0_transpose[0][1]*v_0[i][1])+
                (ab_dotcpm_0_transpose[0][2]*v_0[i][2])+
                (abcpm_0_transpose[0][0]*v_0[i][3])+(abcpm_0_transpose[0][1]*
                v_0[i][4])+(abcpm_0_transpose[0][2]*v_0[i][5]);
        v_1[i][4]=(ab_dotcpm_0_transpose[1][0]*v_0[i][0])+
                (ab_dotcpm_0_transpose[1][1]*v_0[i][1])+
                (ab_dotcpm_0_transpose[1][2]*v_0[i][2])+
                (abcpm_0_transpose[1][0]*v_0[i][3])+(abcpm_0_transpose[1][1]*
                v_0[i][4])+(abcpm_0_transpose[1][2]*v_0[i][5]);
        v_1[i][5]=(ab_dotcpm_0_transpose[2][0]*v_0[i][0])+
                (ab_dotcpm_0_transpose[2][1]*v_0[i][1])+
                (ab_dotcpm_0_transpose[2][2]*v_0[i][2])+
                (abcpm_0_transpose[2][0]*v_0[i][3])+(abcpm_0_transpose[2][1]*
                v_0[i][4])+(abcpm_0_transpose[2][2]*v_0[i][5]);
// Write status information to an external file
        if((j=fopen("statout.001","a"))==NULL)
        {
            fclose(j);
            nerror("unable to open statout.001 in main()");
        }
        if(i<1)
        {
            long double test1,test2,test3,test4,test5,test6,test7,test8,test9;
            long double test10,test11,test12,test13,test14,test15,test16,test17;
            test1=x_p_interpolated/ARCSEC_TO_RAD;
            test2=y_p_interpolated/ARCSEC_TO_RAD;
            test3=-lambda/DEG_TO_RAD;
```

```c
        test4=-epsilon_bar/DEG_TO_RAD;
        test5=delta_psi/ARCSEC_TO_RAD;
        test6=epsilon/DEG_TO_RAD;
        test7=zeta/ARCSEC_TO_RAD;
        test8=-theta/ARCSEC_TO_RAD;
        test9=z/ARCSEC_TO_RAD;
        fprintf(j,"Rotate about the Y1  axis by % 16.13Lf\".\n",test1);
        fprintf(j,"Rotate about the X2  axis by % 16.13Lf\".\n",test2);
        fprintf(j,"Rotate about the Z3  axis by % 16.13Lf degrees.\n",test3);
        fprintf(j,"Rotate about the X4  axis by % 16.13Lf degrees.\n",test4);
        fprintf(j,"Rotate about the Z5  axis by % 16.13Lf\".\n",test5);
        fprintf(j,"Rotate about the X6  axis by % 16.13Lf degrees.\n",test6);
        fprintf(j,"Rotate about the Z7  axis by % 16.13Lf\".\n",test7);
        fprintf(j,"Rotate about the Y8  axis by % 16.13Lf\".\n",test8);
        fprintf(j,"Rotate about the Z9  axis by % 16.13Lf\".\n",test9);
        fprintf(j,"Rotate about the Z10 axis by -0.320288870000 degrees.\n");
        fprintf(j,"Rotate about the Y11 axis by  0.278405860000 degrees.\n");
        fprintf(j,"Rotate about the Z12 axis by -0.320381690000 degrees.\n");

    }
    if(fclose(j)==EOF)
    {
        nerror("unable to close statout.001 in main()");
    }
}
// Write ECI_M50 state vectors to an external file****************************
    if((j=fopen("stateout.001","w"))==NULL)
    {
        fclose(j);
        nerror("unable to open stateout.001 in main()");
    }
    for(i=0;i<NUMBER_OF_VECTORS;i++)
    {
        fprintf(j,"%Lf % .12Le % .12Le % .12Le\n % .12Le % .12Le %.12Le\n",
        t_0[i][2],v_1[i][0],v_1[i][1],v_1[i][2],v_1[i][3],v_1[i][4],v_1[i][5]);
    }
    if(fclose(j)==EOF)
    {
        nerror("unable to close stateout.001 in main()");
    }
// Write ECEF_WGS_84 state vectors in Kalman Filter input file format*********
    if((j=fopen("sv0.001","w"))==NULL)
    {
        fclose(j);
        nerror("unable to open sv0.001 in main()");
    }
```

```c
for(i=0;i<NUMBER_OF_VECTORS;i++)
{

    int minutes;
    long double seconds;
    minutes=(int)(t_0[i][5]/60.0);
    seconds=t_0[i][5]-(60.0*(float)minutes);
    minutes-=((int)(t_0[i][5]/3600.0)*60);
    if(i>0)
    {
        if((t_0[i][0]-t_0[i-1][0])>longest)
        {
            longest=t_0[i][0]-t_0[i-1][0];
            printf("\n%Lf",longest);
        }
    }
    fprintf(j,"%d %5.2Lf % 15.6Lf % 15.6Lf % 15.6Lf % 12.6Lf % 12.6Lf %
12.6Lf\n",
            minutes,seconds,v_0[i][0],v_0[i][1],v_0[i][2],v_0[i][3],
            v_0[i][4],v_0[i][5]);
}
if(fclose(j)==EOF)
{
    nerror("unable to close sv0.001 in main()");
}
}
```

APPENDIX B

```c
/************************************************************************/
/*                                                                    */
/* This program compares state vectors and computes the magnitude of */
/* their position and velocity difference.                           */
/*                                                                    */
/* Author: LT Stephen P. Rehwald, USN                                */
/* Date:   20 March 1994                                         */
/*                                                                    */
/* Functions from "Numerical Recipes in C", Press, W.H., et al,       */
/* Cambridge University Press, 1988 were utilized in this program.    */
/*                                                                    */
/************************************************************************/


/************************************************************************/
/* NUMBER_OF_VECTORS is the number of position/velocity vectors to     */
/* process from the files used as input to this program. Each file     */
/* must, at a minimum, contain this number of vectors. STATEOUT.001 and */
/* REFTRAJ.001 specify the names of the input files. M50DIFFS.001      */
/* specifies the name of the output file.                            */
/************************************************************************/

#define NUMBER_OF_VECTORS 139

#include <math.h>
#include <stdio.h>
#include <stdlib.h>

void nerror(char*);

FILE *j,*k;
int i;
long double t_0[NUMBER_OF_VECTORS][2],v_diff[NUMBER_OF_VECTORS][2];
long double v_0[NUMBER_OF_VECTORS][6],v_1[NUMBER_OF_VECTORS][6];

/*                                                             */
/* nerror is Numerical Recipes standard error handler*/
/*                                                             */

void nerror(char error_text[])
{
    fprintf(stderr,"Numerical Recipes run-time error...\n");
    fprintf(stderr,"%s\n",error_text);
```

```c
        fprintf(stderr,"...now exiting to system...\n");
        _exit(1);
}

void main(void)
{
// Read GPS_M50 and Reference Trajectory state vectors from external files****
    if((j=fopen("stateout.001","r"))==NULL)
    {
        fclose(j);
        nerror("unable to open stateout.001 in main()");
    }
    if((k=fopen("reftraj.001","r"))==NULL)
    {
        fclose(k);
        nerror("unable to open reftraj.001 in main()");
    }
    for(i=0;i<NUMBER_OF_VECTORS;i++)
    {
        fscanf(j,"%Lf %Le %Le %Le %Le %Le %Le\n",&t_0[i][0],&v_0[i][0],
                    &v_0[i][1],&v_0[i][2],&v_0[i][3],&v_0[i][4],&v_0[i][5]);
        fscanf(k,"%Lf %Le %Le %Le %Le %Le %Le\n",&t_0[i][1],&v_1[i][0],
                    &v_1[i][1],&v_1[i][2],&v_1[i][3],&v_1[i][4],&v_1[i][5]);
        //if(i>0)
        {
            if(t_0[i][0]!=t_0[i][1])
            {
                        nerror("time element mismatch in main()");
            }
        }
    }
    if(fclose(j)==EOF)
    {
        nerror("unable to close stateout.001 in main()");
    }
    if(fclose(k)==EOF)
    {
        nerror("unable to close reftraj.001 in main()");
    }
// Main loop to calculate time, rotation matrices, and transformations********
    for(i=0;i<NUMBER_OF_VECTORS;i++)
    {
// Compute the magnitude of the position difference**************************
        v_diff[i][0]=sqrtl(powl((v_1[i][0]-v_0[i][0]),2.0)+powl((v_1[i][1]-
                                            v_0[i][1]),2.0)+powl((v_1[i][2]-
v_0[i][2]),2.0));
```

83

```c
// Compute the magnitude of the velocity difference***************************
      v_diff[i][1]=sqrtl(powl((v_1[i][3]-v_0[i][3]),2.0)+powl((v_1[i][4]-
                                   v_0[i][4]),2.0)+powl((v_1[i][5]-
v_0[i][5]),2.0));
      }
// Write differences to an external file*****************************************
   if((j=fopen("m50diffs.001","w"))==NULL)
   {
      fclose(j);
      nerror("unable to open DIFFERENCE_FILE in main()");
   }
   for(i=0;i<NUMBER_OF_VECTORS;i++)
   {
      fprintf(j,"%Lf %16.13Lf
%16.13Lf\n",t_0[i][0],v_diff[i][0],v_diff[i][1]);
   }
   if(fclose(j)==EOF)
   {
      nerror("unable to close DIFFERENCE_FILE in main()");
   }
}
```

APPENDIX C

The plots appearing in this appendix depict the number of usable GPS satellites being tracked by the recevier. If more than three usable satellites are being tracked, the indication is given either that the receiver is doing position fixes, or that PDOP is too high. VDOP, HDOP, and PDOP are also shown (PDOP is a combination of VDOP and HDOP) at the corresponding times. The segment at the center (right to left) of each plot corresponds to the data chosen for analysis in Chapter III, Table 3. For comparison purposes the times shown in Figures 5-10 equate to times appearing on the TANSGRAPH plots as follows:

- 6261.434514 hours is FRI:21:26:04.25
- 6261.507708 hours is FRI:21:30:27.75

- 6261.893542 hours is FRI:21:53:36.75
- 6261.988264 hours is FRI:21:59:17.75

- 6265.089236 hours is SAT:01:05:21.25
- 6265.193264 hours is SAT:01:11:35.75

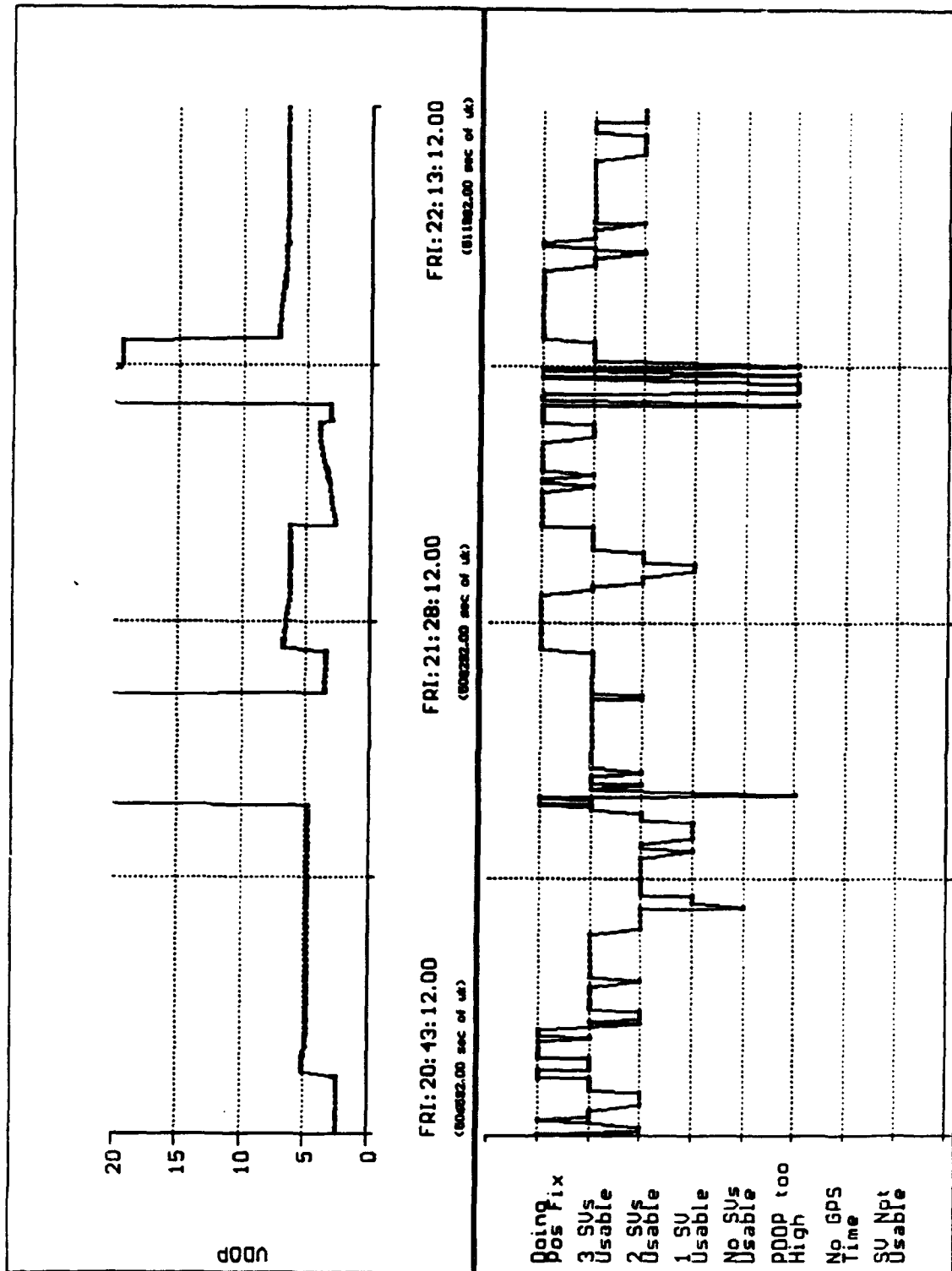**Figure C-1: File P.004 TANSGRAPH Plot Showing No. of Usable Satellites**
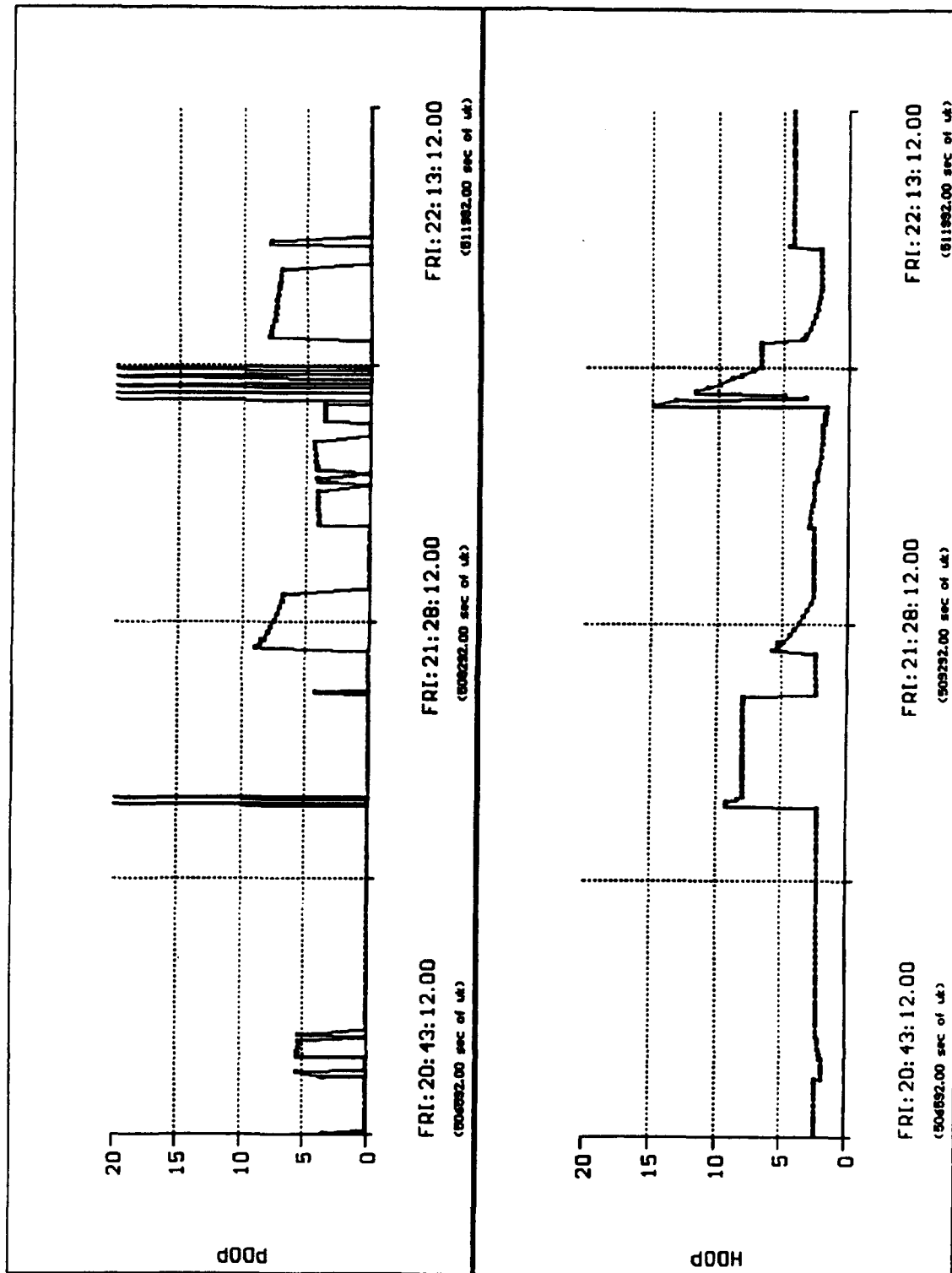
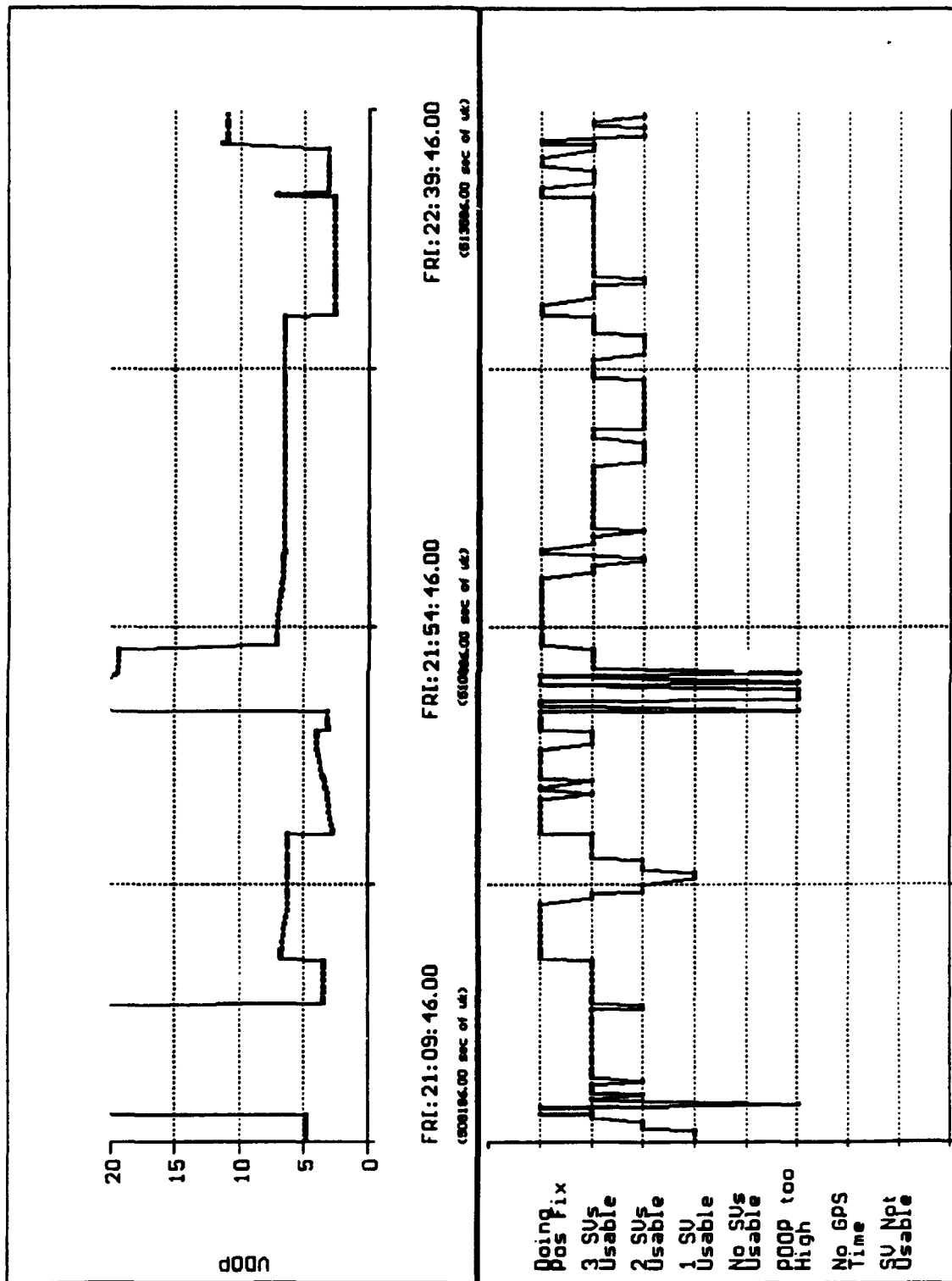**Figure C-2: File P.004 TANSGRAPH Plot Showing PDOP**

87

Figure C-3: File P.005 TANSGRAPH Plot Showing No. of Usable Satellites
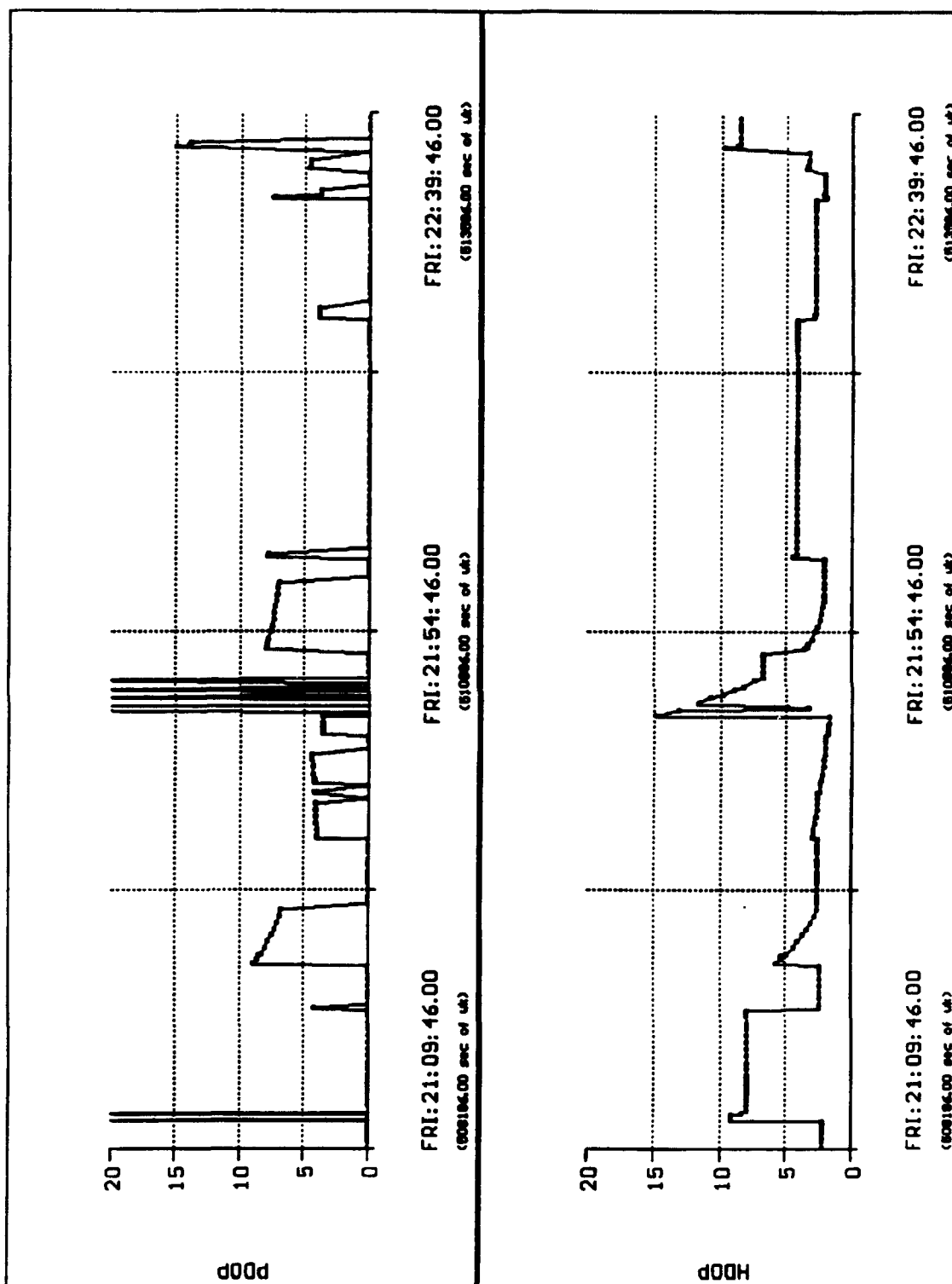
Figure C-4: File P.005 TANSGRAPH Plot Showing PDOP
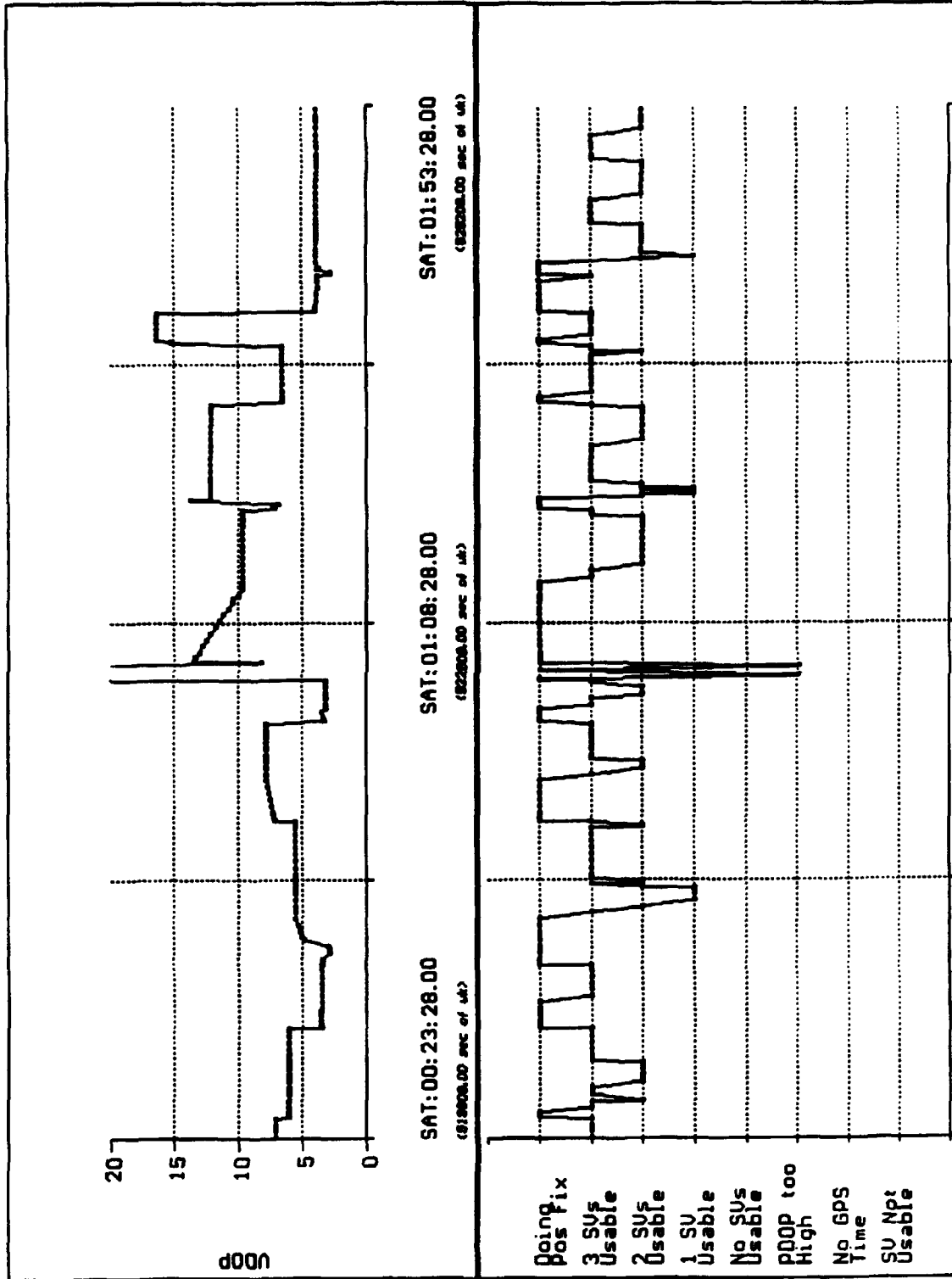
Figure C-5: File P.008 TANSGRAPH Plot Showing No. of Usable Satellites

90
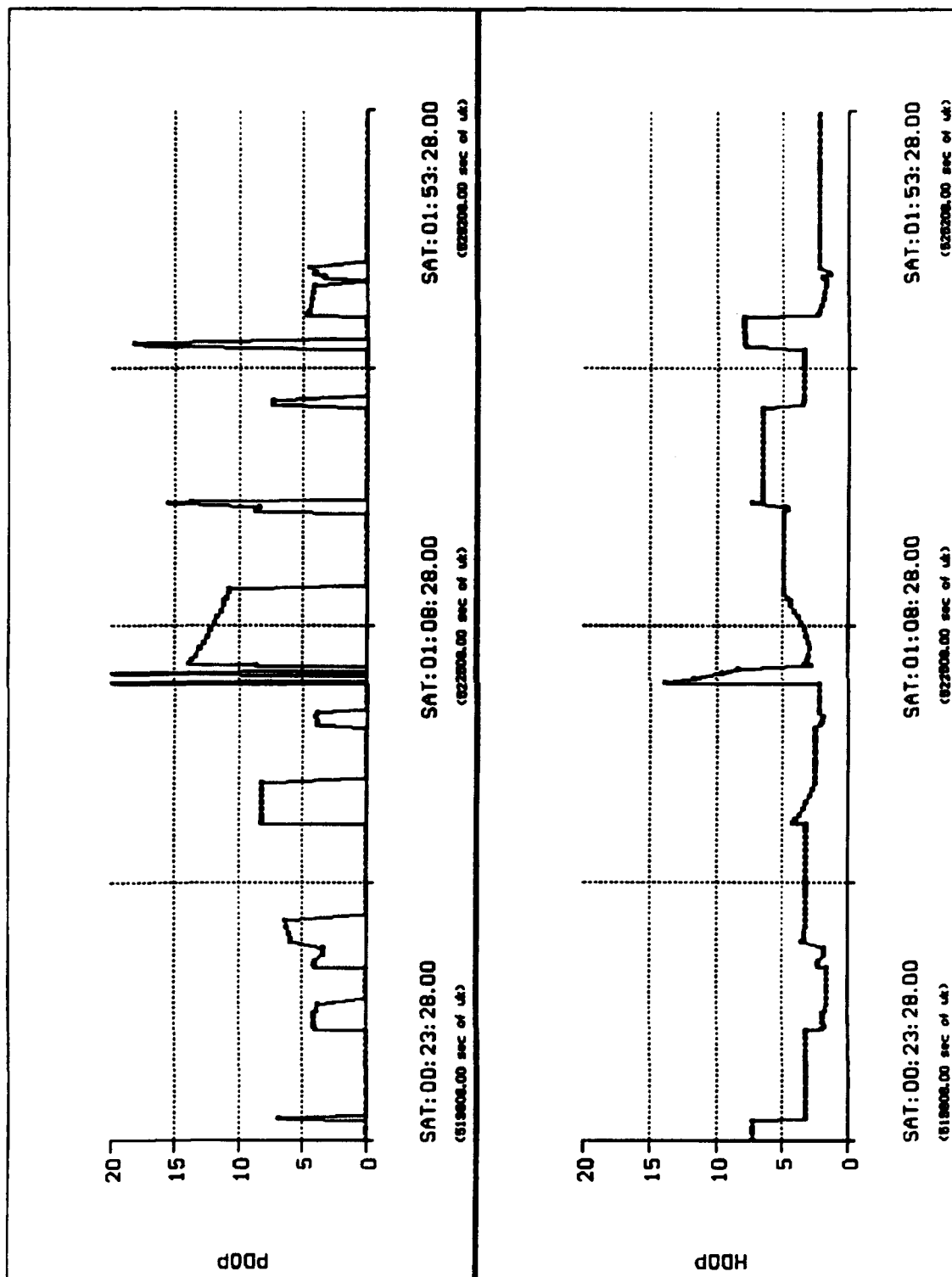
Figure C-6: File P.008 TANSGRAPH Plot Showing PDOP

91

APPENDIX D

This Kalman Filter is designed for a user-specified number of data points which, in this case, is found in a file named, flttest2.dat in ASCII format. Flttest2.dat is included in this appendix. It can be found beginning on the third page of Appendix D. The function, getvals(), reads the flttest2.dat into the appropriate vectors: tsec, X, XD, Y, YD, Z, and ZD. This short function is included at the end of the filter program.

```
***************************************************************
load flttest2.dat

[tsec, X, XD, Y, YD, Z, ZD] = getvals( i, flttest2);

%Initialize variables

A=[0 1; 0 0];
B=[0 1]';
I=eye(2);
C=[1 0;0 1];
Tf=4;
dt=1;
[Phi,Del]=c2d(A,B,dt);
Pkkm1=1e6*I;
R=I;
Q=.01*I;
kmax=72 ;
u=zeros(2,kmax);

xkk=zeros(2,kmax+1);
xkkm1=zeros(2,kmax+1);
ykk=zeros(2,kmax+1);
ykkm1=zeros(2,kmax+1);
zkk=zeros(2,kmax+1);
zkkm1=zeros(2,kmax+1);

xhat=zeros(2,kmax+1);
yhat=zeros(2,kmax+1);
zhat=zeros(2,kmax+1);
time=zeros(1,kmax);
t=zeros(1,kmax);
g=zeros(4,kmax);
```

```
%Begin the filtering process
for (i=1:kmax)
   Phi(1,2)=tsec(i+1)-tsec(i);
   G=Pkkm1*C'*inv(C*Pkkm1*C' +R);
   Pkk=(I-G*C)*Pkkm1;
   Pkkm1=Phi*Pkk*Phi' + Q;


   xkk(:,i) = xkkm1(:,i) +G*([X(i,1) XD(i,1)]'-C*xkkm1(:,i));
   xkkm1(:,i+1)=Phi*xkk(:,i);


   ykk(:,i)= ykkm1(:,i) +G*([Y(i,1) YD(i,1)]'-C*ykkm1(:,i));
   ykkm1(:,i+1)=Phi*ykk(:,i);


   zkk(:,i)= zkkm1(:,i) +G*([Z(i,1) ZD(i,1)]'-C*zkkm1(:,i));
   zkkm1(:,i+1)=Phi*zkk(:,i);


   xhat(:,i) = C*xkk(:,i);
   yhat(:,i) = C*ykk(:,i);
   zhat(:,i) = C*zkk(:,i);


   t(i)=time(i);


   time(i+1)= time(i) + dt;
   g(1,i)=G(1,1);
   g(2,i)=G(2,1);
   g(3,i)=G(1,2);
   g(4,i)=G(2,2);
end;



********************************************************************
%This getvals() function works for 98 elements
function [tsec, X, XD, Y, YD, Z, ZD] = getvals( j , flttest2)
i =1: 98;
tsec = 60* flttest2(i, 1) + flttest2(i, 2);
X = flttest2(i, 3);
XD = flttest2(i, 6);
Y = flttest2(i, 4);
YD = flttest2(i, 7);
Z = flttest2(i, 5);
ZD = flttest2(i, 8);
********************************************************************
```

93

Time (Min/Sec)		X	Y	Z	Xdot	Ydot	Zdot
26	04.250	-6671.363	276.288	196.336	-0.384	-6.293	-3.675
26	06.750	-6672.305	260.552	187.146	-0.364	-6.294	-3.676
26	09.250	-6673.186	244.821	177.958	-0.344	-6.294	-3.676
26	11.750	-6674.019	229.087	168.770	-0.324	-6.295	-3.677
26	14.250	-6674.797	213.356	159.582	-0.304	-6.296	-3.677
26	16.750	-6675.528	197.622	150.391	-0.284	-6.296	-3.678
26	19.750	-6676.331	178.744	139.364	-0.260	-6.297	-3.678
26	22.250	-6676.958	163.002	130.168	-0.240	-6.297	-3.679
26	24.750	-6677.533	147.260	120.970	-0.220	-6.298	-3.679
26	29.250	-6678.442	118.921	104.413	-0.184	-6.298	-3.680
26	33.750	-6679.185	90.579	87.855	-0.148	-6.299	-3.680
26	36.250	-6679.531	74.833	78.653	-0.128	-6.299	-3.680
26	38.750	-6679.821	59.087	69.453	-0.108	-6.299	-3.681
26	40.750	-6680.022	46.489	62.092	-0.092	-6.299	-3.681
26	43.250	-6680.238	30.737	52.887	-0.072	-6.299	-3.681
26	45.250	-6680.367	18.138	45.524	-0.055	-6.299	-3.681
26	47.750	-6680.485	2.389	36.320	-0.035	-6.299	-3.681
26	49.750	-6680.535	-10.207	28.960	-0.019	-6.299	-3.681
26	52.250	-6680.556	-25.952	19.758	0.001	-6.299	-3.681
26	54.250	-6680.532	-38.549	12.398	0.017	-6.299	-3.681
26	57.750	-6680.425	-60.594	-0.486	0.045	-6.298	-3.681
27	00.250	-6680.295	-76.343	-9.692	0.065	-6.298	-3.681
27	06.250	-6679.761	-114.129	-31.778	0.113	-6.297	-3.681
27	09.250	-6679.394	-133.025	-42.823	0.137	-6.297	-3.681
27	11.750	-6679.024	-148.767	-52.023	0.157	-6.296	-3.680
27	13.750	-6678.699	-161.361	-59.384	0.173	-6.296	-3.680
27	16.250	-6678.238	-177.096	-68.585	0.193	-6.295	-3.680
27	18.750	-6677.734	-192.836	-77.787	0.213	-6.295	-3.680
27	21.250	-6677.175	-208.571	-86.986	0.233	-6.294	-3.679
27	23.750	-6676.566	-224.307	-96.183	0.253	-6.294	-3.679
27	26.250	-6675.912	-240.042	-105.382	0.273	-6.293	-3.679
27	28.750	-6675.208	-255.775	-114.580	0.293	-6.292	-3.678

Time (Min/Sec)		X	Y	Z	Xdot	Ydot	Zdot
27	31.250	-6674.447	-271.501	-123.775	0.313	-6.291	-3.678
27	33.750	-6673.641	-287.226	-132.969	0.333	-6.290	-3.678
27	36.250	-6672.785	-302.953	-142.162	0.353	-6.290	-3.677
27	38.750	-6671.874	-318.676	-151.352	0.373	-6.289	-3.677
27	42.250	-6670.519	-340.684	-164.219	0.401	-6.287	-3.676
27	44.750	-6669.493	-356.401	-173.406	0.421	-6.286	-3.675
27	47.250	-6668.419	-372.117	-182.595	0.441	-6.285	-3.675
27	49.750	-6667.290	-387.827	-191.780	0.461	-6.284	-3.674
27	52.250	-6666.105	-403.533	-200.963	0.481	-6.283	-3.673
27	54.750	-6664.890	-419.244	-210.150	0.501	-6.282	-3.673
27	57.250	-6663.604	-434.943	-219.329	0.521	-6.280	-3.672
27	59.750	-6662.278	-450.645	-228.506	0.541	-6.279	-3.671
28	02.250	-6660.909	-466.343	-237.688	0.561	-6.278	-3.670
28	08.250	-6657.398	-503.999	-259.701	0.608	-6.275	-3.668
28	12.250	-6654.903	-529.091	-274.373	0.640	-6.272	-3.667
28	15.250	-6652.943	-547.904	-285.371	0.664	-6.270	-3.666
28	17.750	-6651.254	-563.576	-294.533	0.684	-6.269	-3.665
28	20.250	-6649.531	-579.252	-303.699	0.704	-6.267	-3.664
28	22.750	-6647.752	-594.920	-312.860	0.724	-6.265	-3.663
28	25.250	-6645.921	-610.583	-322.017	0.744	-6.264	-3.662
28	27.750	-6644.041	-626.239	-331.173	0.764	-6.262	-3.661
28	30.250	-6642.106	-641.891	-340.322	0.783	-6.260	-3.660
28	32.250	-6640.516	-654.407	-347.636	0.799	-6.259	-3.659
28	35.250	-6638.075	-673.175	-358.607	0.823	-6.256	-3.657
28	37.750	-6635.992	-688.814	-367.748	0.843	-6.254	-3.656
28	40.250	-6633.862	-704.448	-376.888	0.863	-6.252	-3.655
28	42.250	-6632.117	-716.949	-384.196	0.879	-6.251	-3.654
28	44.750	-6629.896	-732.575	-393.329	0.899	-6.249	-3.652
28	48.250	-6626.698	-754.440	-406.107	0.926	-6.246	-3.651
28	50.750	-6624.355	-770.051	-415.231	0.946	-6.244	-3.649
28	53.250	-6621.958	-785.653	-424.350	0.966	-6.241	-3.648
28	55.750	-6619.530	-801.257	-433.471	0.985	-6.239	-3.647
28	58.250	-6617.038	-816.852	-442.583	1.005	-6.237	-3.645

95

Time (Min/Sec)	X	Y	Z	Xdot	Ydot	Zdot
29 00.750	-6614.496	-832.438	-451.693	1.025	-6.235	-3.644
29 02.750	-6612.430	-844.904	-458.979	1.040	-6.233	-3.643
29 08.250	-6606.585	-879.167	-479.003	1.084	-6.227	-3.639
29 10.750	-6603.849	-894.732	-488.099	1.104	-6.225	-3.637
29 13.250	-6601.072	-910.294	-497.193	1.123	-6.222	-3.636
29 15.750	-6598.238	-925.844	-506.282	1.143	-6.219	-3.634
29 18.250	-6595.336	-941.380	-515.359	1.163	-6.217	-3.632
29 21.250	-6591.818	-960.028	-526.254	1.186	-6.214	-3.630
29 23.750	-6588.830	-975.561	-535.328	1.206	-6.211	-3.629
29 26.250	-6585.793	-991.084	-544.398	1.226	-6.208	-3.627
29 28.750	-6582.693	-1006.595	-553.459	1.245	-6.205	-3.625
29 31.250	-6579.563	-1022.109	-562.520	1.265	-6.202	-3.623
29 33.250	-6577.019	-1034.511	-569.768	1.281	-6.200	-3.622
29 35.750	-6573.782	-1050.003	-578.816	1.300	-6.197	-3.620
29 38.250	-6570.512	-1065.497	-587.865	1.320	-6.194	-3.618
29 42.250	-6565.169	-1090.261	-602.328	1.351	-6.189	-3.615
29 44.250	-6562.460	-1102.640	-609.560	1.367	-6.187	-3.613
29 46.750	-6559.027	-1118.105	-618.592	1.386	-6.183	-3.611
29 49.250	-6555.538	-1133.561	-627.619	1.406	-6.180	-3.609
29 51.750	-6551.994	-1149.003	-636.639	1.425	-6.177	-3.607
29 54.750	-6547.685	-1167.531	-647.456	1.449	-6.173	-3.604
29 56.750	-6544.768	-1179.872	-654.661	1.464	-6.170	-3.603
29 59.250	-6541.083	-1195.293	-663.665	1.484	-6.167	-3.600
30 01.250	-6538.093	-1207.625	-670.860	1.499	-6.164	-3.599
30 03.250	-6535.081	-1219.949	-678.059	1.515	-6.161	-3.597
30 05.750	-6531.356	-1235.345	-687.049	1.534	-6.158	-3.595
30 11.250	-6522.726	-1269.198	-706.810	1.577	-6.150	-3.590
30 15.250	-6516.351	-1293.782	-721.160	1.608	-6.144	-3.586
30 17.750	-6512.313	-1309.144	-730.124	1.628	-6.140	-3.583
30 20.250	-6508.208	-1324.482	-739.077	1.647	-6.137	-3.581
30 22.250	-6504.904	-1336.754	-746.237	1.662	-6.134	-3.579
30 24.750	-6500.726	-1352.086	-755.180	1.682	-6.130	-3.576
30 27.750	-6495.650	-1370.469	-765.907	1.705	-6.125	-3.573

LIST OF REFERENCES

Defense Mapping Agency, Report 8350.2, *Department of Defense World Geodetic System 1984. Its Definition and Relationships with Local Geodetic Systems*, pp.2-1, 1987.

Handout from AA 3276, Introduction to Avionics, class taught by Dr. I.I. Kaminar, "Design and performance analysis of Kalman filters," Naval Postgraduate School, September 1993.

Kachmar, P.M., Chu, W., Neirinckx, P.and Montez, M.,"U.S. Space Shuttle: Integrated GPS Navigation Capability," *Proceedings of ION GPS-93*, Sixth International Technical Meeting, Salt Lake City, Utah, September 1993, pp.313-326.

Leewen, A.V., Rosen, E., and Carrier, L., The Global Positioning System and Its Application in Spacecraft Navigation, *Journal of the Institute of Navigation*, pp.118-135, Summer 1979.

Naval Postgraduate School, *Advanced Avionics Technology*, Kaminar, I.I., (Principle Investigator), pp.117-138 and pp.167-169.

NAVSTAR-GPS Joint Program Office, *NAVSTAR GPS User Equipment Introduction* (For Official Use Only), pp.3-1 to 3-9 and pp.8-1 to 8-7, U.S. Air Force Space Systems Division, 1990.

Milliken, R.J., and Zoller, C.J., "Principle of Operation of NAVSTAR and System Characteristics, " *Global Positioning System*, Vol. I, Washington, D.C., Institute of Navigation, 1980.

Saunders, P.E.and others, "The First Flight Tests of GPS on the Space Shuttle," paper presented at the Institute of Navigation Convention, San Diego, California, 24 January 1994.

BIBLIOGRAPHY

The Astronomical Almanac for the Year..., Washington: Government Printing Office.

Chaffee, J.W., Zare, K., and Pojman, J., "Autonomous Navigation for COMET with GPS," *Proceedings of ION GPS-93*, Sixth International Technical Meeting, Salt Lake City, Utah, September 1993, pp.293-297.

Duiven, E.M., Connolly, P.E., and Alfriend, K.T., "Satellite Tracking Using GPS Measurements," IEEE, 1978.

Hashida, Y., and Unwin, M.J., "Satellite Attitude From a Single GPS Antenna," *Proceedings of ION GPS-93*, Sixth International Technical Meeting, Salt Lake City, Utah, September 1993, pp.355-363.

Hofmann-Wellenhof, B., Lichtenegger, H., and Collins, J., *Global Positioning System Theory and Practice*, Second Edition, Springer-Verlag Wien New York, 1993.

McCarthy, D., Carter, M.S., and Luzum, B.J., *Operational Prediction of the Earth's Orientation in the International Earth Rotation Service (IERS) Reference Frame*, U.S. Naval Observatory, Washington, D.C., c.1991.

NASA Technical Memorandum, Report X-58153, *Coordinate Systems for the Space Shuttle Program*, October 1974.

Notes and data from Ed Brown of Rockwell Shuttle Navigation at NASA Johnson Space Center.

Notes and data from Penny Saunders of NASA Johnson Space Center.

Seidelmann, K.P.(editor), *Explanatory Supplement to the Astronomical Almanac (Rev. Ed.)*, Mill Valley, CA: University Science Books, 1992.

Taff, L.G., *Computational Spherical Astronomy*, New York: Wiley, 1981.

Tang, W., and Howell, G., "Integrated GPS/INS Kalman Filter Implementation Issues," *Proceedings of ION GPS-93*, Sixth International Technical Meeting, Salt Lake City, Utah, September 1993, pp.217-224.

TANS QUADREX Space Configuration GPS Receiver Interface Control Document, Part Number 17035-SP, Revision E, Trimble Navigation Limited, Sunnyvale, CA, 1992.

Zyla, L.V., and Montez, M.N., "Use of Two GPS Receivers in Order to Perform Space Vehicle Orbital Rendezvous," *Proceedings of ION GPS-93*, Sixth International Technical Meeting, Salt Lake City, Utah, September 1993, pp.301-312.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center   2
   Cameron Station
   Alexandria, Virginia 22304-6145

2. Dudley Knox Library   2
   Naval Postgraduate School
   Code 52
   Monterey, California 93943-5002

3. CDR Randy Wight   1
   Naval Postgraduate School
   Code SP/Wt
   Monterey, California 93943-5002

4. Professor I. M. Ross   1
   Naval Postgraduate School
   Code AA/RO
   Monterey, California 93943-5002

5. Professor Don Danielson   1
   Naval Postgraduate School
   Code MA/DD
   Monterey, California 93943-5002

6. Commander   1
   Naval Space Command
   Dahlgren, Virginia 22448

7. LT Lee Barker   1
   SPAWAR 40
   2451 Crystal Dr.
   Arlington, Virginia 22245-5200

8. Dr. Jim Newman   1
   NASA Johnson Space Center
   Mail Code CB
   Houston, Texas 77058

9. Flora Lowes   1
   NASA Johnson Space Center
   Mail Code DM4
   Houston, Texas 77058

10. Penny Saunders   1
    NASA Johnson Space Center
    Mail Code EE6
    Houston, Texas 77058

11. Stephen Rehwald   1
    P.O. Box 1129
    San Bernardino, California 92402-1129

12. Carolyn Tyler   1
    122½ Dunecrest Ave.
    Monterey, California 93940